7-2006

Placement, visibility and coverage analysis of dynamic pan/tilt/zoom camera sensor networks

John A. Ruppert

Placement, Visibility and Coverage Analysis of Dynamic Pan/Tilt/Zoom Camera Sensor Networks

by

John A. Ruppert

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Engineering

Supervised by

Assistant Professor Dr. Shanchieh Jay Yang
Department of Computer Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York

July 2006

Approved By:

Shanchieh Jay Yang

Dr. Shanchieh Jay Yang
Assistant Professor
Primary Adviser

Andreas Savakis

Dr. Andreas Savakis
Professor and Department Head, Department of Computer Engineering

Chris M. Homan

Dr. Chris Homan
Assistant Professor, Department of Computer Science

Thesis Release Permission Form

Rochester Institute of Technology

Kate Gleason College of Engineering

Title: Placement, Visibility and Coverage Analysis of Dynamic Pan/Tilt/Zoom Camera Sensor Networks

I, John A. Ruppert, hereby grant permission to the Wallace Memorial Library reporduce my thesis in whole or part.

John Ruppert

John A. Ruppert

7-28-06

Date

Dedication

To my parents, James and Lori Ruppert.

Acknowledgments

I would like to thank Dr. Shanchieh Jay Yang for his continued support and encouragement throughout the course of this work. I would also like to thank Dr. Andreas Savakis for inspiring my interest in camera networking research and Dr. Chris Homan for lending his insight in computational geometry to help understand and formulate the problem statement.

Abstract

Multi-camera vision systems have important application in a number of fields, including robotics and security. One interesting problem related to multi-camera vision systems is to determine the effect of camera placement on the quality of service provided by a network of Pan/Tilt/Zoom (PTZ) cameras with respect to a specific image processing application. The goal of this work is to investigate how to place a team of PTZ cameras, potentially used for collaborative tasks, such as surveillance, and analyze the dynamic coverage that can be provided by them.

Computational Geometry approaches to various formulations of sensor placement problems have been shown to offer very elegant solutions; however, they often involve unrealistic assumptions about real-world sensors, such as infinite sensing range and infinite rotational speed. Other solutions to camera placement have attempted to account for the constraints of real-world computer vision applications, but offer solutions that are approximations over a discrete problem space.

A contribution of this work is an algorithm for camera placement that leverages Computational Geometry principles over a continuous problem space utilizing a model for dynamic camera coverage that is simple, yet representative. This offers a balance between accounting for real-world application constraints and creating a problem that is tractable.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recently, there has been increased emphasis on video surveillance research for various purposes, *e.g.*, to avoid theft in commerce and national security. Video surveillance systems typically consist of multiple video cameras which are used to monitor an environment. Traditional video surveillance systems have been controlled by human operators or have used regular camera rotation schemes to observe objects moving within the field of view of the cameras. As networking and image processing research progresses, more advanced systems are being developed to automate the process of video surveillance. This involves the development of networking policies and image processing and computer vision algorithms to detect, locate and track an object as it moves through an environment.

An important concept of multi-camera systems is measuring the quality of service it provides. This could include, for example, how well a network of camera sensors monitors an area or how effectively it can detect intruders.

The focus of this work is to investigate the coverage provided by a network of Pan/Tilt/Zoom cameras. A PTZ camera is one that has motorized control for rotating the camera in all directions as well as zooming in on an image. This work attempts to pair up a set of PTZ cameras with an image processing application to see the effects on coverage of the camera set when taking into account the constraints of a specific application.

The question this work addresses is: How may one place a network of PTZ cameras, potentially used for collaborative tasks, such as video surveillance, and analyze the dynamic coverage that can be provided by them?

The key challenge here is that PTZ cameras may adjust their pan, tilt and zoom levels over time and cooperate in order to improve network coverage. This work does not attempt to provide an optimal solution to the problem posed above, but rather the goal is to develop a reasonable approach to gain insight into the problem and learn from experience. The results of this work include:

1. A study of the related work
2. Static and dynamic models for camera coverage
3. An algorithm for automated camera placement that places cameras in a room that provides complete coverage of the area of interest
4. An algorithm for camera visibility to deal with occlusion of walls within a room and to compute the overlapping coverage
5. A simulated environment and coverage metrics for implementing and testing camera coverage algorithms
6. Camera placement and parameter adjustment strategies used to target a particular coverage goal

1.1 Related Work

At first glance of the problem, it is unclear how to model dynamic coverage for a PTZ camera, not to mention gaining insight on how to place cameras that have dynamic coverage capabilities. So, a starting point for this research was to determine what it means to model "coverage". This section summarizes the related work study, highlights important concepts that are drawn upon and presents an argument for the importance of this thesis.

1.1.1 Sensor Networks

The emergence of sensor networks as a premier research topic is a key motivating factor for this work. Sensor networks have been made possible by recent advances in communications and embedded micro-sensing technologies. A sensor network consists of a number of sensors nodes that collaborate to perform a larger sensing task and to communicate the information to a sink, or a central location [2]. Sensor nodes are small electronic devices equipped with one or more sensors, a transceiver, storage components, processing components and possibly an actuator. The low-cost, low-power, multi-function capabilities of sensor nodes enables unique features, such as random deployment and collaborative sensing. These features allow for a wide range of potential applications and thus make sensor networks an interesting topic for research. Exemplary applications include battlefield surveillance, forest fire detection, smart environments and inventory control. This work considers a network of PTZ camera sensors targeting image processing applications.

1.1.2 Coverage: Examples from other fields

Coverage is an important concept that will be drawn upon in this work. The term "coverage" typically refers to measures of effectiveness which are used to characterize the overall performance of a system. The idea of "coverage" appears in a number of applications in various research fields, including the Art Gallery Problem [24], satellite ocean monitoring [13] and the coordinated positioning of mobile robots [12].

The Art Gallery Problem involves determining the number and locations of observers needed to monitor the interior of an n-wall art gallery room such that every point in the room is seen by at least one observer. Using computational geometry, it has been shown that $\lfloor \frac{n}{3} \rfloor$ observers is occasionally necessary and always sufficient to guard an art gallery with n vertices in its polygonal representation. Quite often not all of the $\lfloor \frac{n}{3} \rfloor$ observers are needed. For example, consider the two polygons in Figure 1.1 each with 12 vertices, one which requires $\lfloor \frac{n}{3} \rfloor$ guards and one which does not.

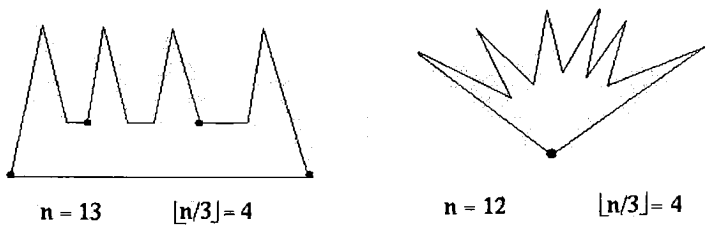


Figure 1.1: Observer placement for the Art Gallery Problem.

The question then becomes: How can one determine algorithms for minimum guard placement? Research has shown this type of problem to be NP-hard [24].

In [13], satellites are used for coverage of global oceans to determine the abundance of phytoplankton. The authors of this work define "coverage" as percentage of the ocean surface that can be monitored in a single day. A method is presented to improve ocean coverage by exploiting redundancy in information from various missions. This is done by collecting and merging information from satellites at various orbits.

The concept of "coordinated positioning and movement in concert" for command control of many-robot systems is considered in [12]. Here, "coverage" refers to maintaining a spatial relationship between mobile robots with respect to a specific mission goal. The author defines three types of coverage: (1) blanket coverage, (2) barrier coverage and (3) sweep coverage. Figure 1.2 illustrates these coverage behaviors.

Blanket coverage refers to determining a static arrangement of elements that maximizes the total amount of area covered. Barrier coverage also involves a static arrangement of elements, but its goal is to minimize the probability that penetration through the barrier will go undetected. Lastly, the objective of sweep coverage is to move a group of elements across an area of interest taking into account the trade off between maximizing the area covered and minimizing the number of missed detections per unit area.

1.1.3 Coverage in Sensor Networks

Coverage is also an important concept in sensor networks and, since the applications being considered in this work fit under the category of sensor networks, this section presents in

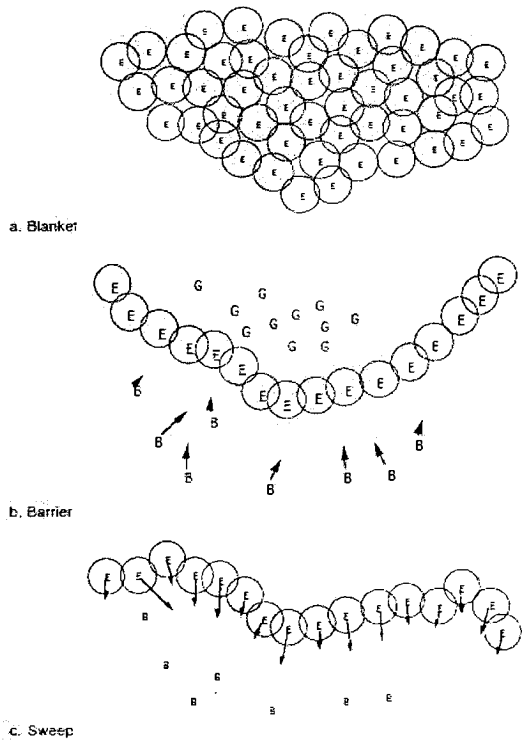


Figure 1.2: Coverage Behaviors. E, G, and B represent system Elements,"Good guy" to be protected, and "Bad guys" to be engaged, respectively. The circles around system elements represent the effective sensor/effector engagement radius. [12]

great detail how coverage has been defined for sensor networks.

The coverage problem in sensor networks is centered around quantifying the quality with which sensors monitor a physical space. Due to the variety of available sensors and the numerous applications of sensor networks, many different formulations of the coverage problem have been proposed by the research community. The most studied types of sensor network coverage include the following subjects: (1) point coverage, (2) area coverage and (3) barrier coverage. Figure 1.3 illustrates each of these concepts.

The following section presents some typical models used for sensing and discusses in detail the types of coverage illustrated in Figure 1.3. In addition, some examples of coverage formulations from the literature are presented here.

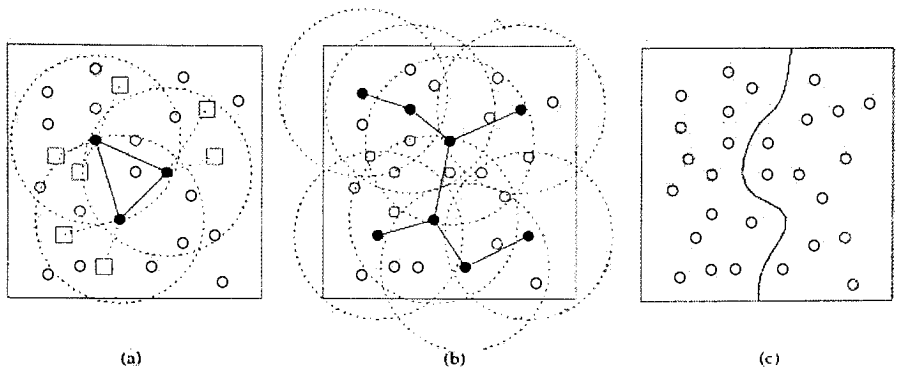


Figure 1.3: (a) point coverage, (b) area coverage, (c) barrier coverage [7]

Sensor Models

Many formulations of the coverage problem use sensor models as a starting point for the definition of network coverage. The most basic sensor model proposed in the literature is a disk model where the radius of the disk represents the sensing range of the sensor [15]. This type of model will be referred to as the 0-1 sensor model throughout this work. Figure 1.4 illustrates the 0-1 sensor model for sensors with both uniform and non-uniform sensing ranges.

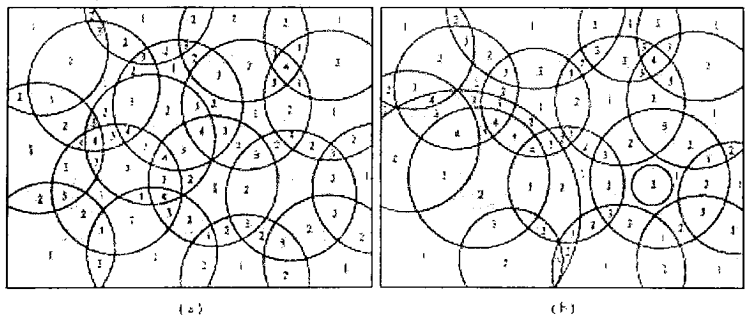


Figure 1.4: Examples of 0-1 sensor model: (a) uniform disks and (b) non-uniform disks. [15]

More complex senor models that take into account the uncertainty of sensors due to various factors are considered in [21], [19] and [1]. Though the details of the mathematical representations vary, the sensor models defined in these works all have in common the idea

that sensing ability diminishes as the distance between a sensor and a target increases, thus a sensor is more likely to detect an event happening at a closer distance as opposed to a distance further away. This will be referred to as the probabilistic sensor model. Equation 1.1 is an example of how such a sensor model is formulated mathematically.

$$S(s,p) = \frac{\lambda}{[d(s,p)]^K} \tag{1.1}$$

Here, $d(s,P)$ is the Euclidean distance between the sensor s and the point p, and positive constants λ and K are sensor technology-dependent parameters.

Deployment

In addition to the sensor model used, another important consideration for analysis of network coverage is how sensors are deployed in an environment. Two types of deployment are defined in the literature: (1) deterministic and (2) random. Deterministic coverage refers to coverage provided by a set of sensor nodes deployed according to a predefined layout. The deployment can be uniform or weighted to account for areas of increased network activity, or hot spots. Examples of uniform patterns include repeated geometric patterns, such as square, triangle and hexagon. In this case, coverage analysis simplifies to the analysis of one basic cell because of the symmetric properties of the uniform pattern. As an example of weighted deterministic coverage, consider camera coverage of valuable items in a museum. To ensure protection of more valuable items, an increased number of cameras could be used for monitoring them.

It is not always feasible, however, to place sensors in a deterministic fashion. Consider placing nodes in a battlefield or a disaster area for instance. In such cases it is important to consider random deployments. An example of assumptions made in the related work is that nodes are uniformly distributed on a 2-D area.

Point-based and Area-based Coverage

The objective of point-based coverage is to achieve a static arrangement of sensor nodes such that each point in a discrete set of points-of-interest is covered by at least one sensor node at any point in time. Figure 1.3 (a) illustrates the goal of point coverage. The set of points-of-interest that are required to be covered are depicted as squares and the sensor nodes are depicted as circles. The connected black nodes are active nodes and can adequately cover the target points. A scheduling algorithm for consumption of energy based on rotating which nodes are active can be used to determine which nodes should be active at any given time in order to meet coverage requirements.

Area coverage is an extension of point-based coverage and focuses on monitoring a particular area or continuous set of points in the physical world. Figure 1.3 (b) illustrates the area coverage concept. The black sensor nodes are used to represent the active sensor nodes and the target area to be covered is the large box. The active nodes ensure that every point within the target area is covered, as opposed to Figure 1.3 (a) in which one is only concerned with covering a few discrete points.

The analysis of point-based coverage is also applicable to area-based coverage, and vice versa. The only difference is the subject to be covered. Next, some example formulations for point-based and area-based coverage of sensor networks with both deterministic and random deployments are presented. These examples use the previously defined 0-1 sensor model for coverage analysis.

If the placement of sensor nodes can be achieved exactly as planned, the coverage problem essentially means planning the placement of nodes to meet desired coverage requirements. In this case, the goal of coverage can be, for example, determining the minimum number of sensors needed to cover a given discrete set of points such that every target point is covered by at least one sensor, as considered by the authors in [3].

In [6], the authors consider a set of sensor nodes that are randomly deployed to monitor the target points-of-interest. Similarly to [3], the goal of coverage is to ensure that at all times, every target point is covered by at least one sensor; however, since the sensor node

formation is not known, the algorithm for maintaining coverage is more complex. This is accomplished by defining disjoint sets of sensor nodes such that every set of sensors nodes provides complete coverage of the target points. In order to preserve energy, only one of the sets is active at any given time and the sets are activates in a sequential order. The algorithm aims to determine the maximum number of disjoint sets so that the time between activation of a particular sensor is thus also maximized.

In addition to their point based coverage analysis, the authors in [3] also consider placing a minimum number of sensor nodes to cover a target area or continuous set of points. The authors define the concept of an r-strip, or a string of 0-1 sensors along a line and use it to analyze performance ratios for bounded convex regions. This concept is illustrated in Figure 1.5. For more details, see [3].

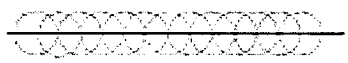


Figure 1.5: r-strip [3]

An important consideration for sensor network design is fault tolerance. Sensor nodes may fail due to lack of power, physical damage or environmental interference, thus there is a need for redundancy or overlap of sensing areas. Many different applications require varying degrees of redundancy. Coverage analysis provides a methodology in which redundancy can be quantified. This idea is presented in terms of a coverage formulation for random area coverage in [15]. The authors define the term k-coverage to refer to the coverage degree where k represents the number of sensor nodes that cover a particular area. The higher the value of k, the more robust the network is to sensor failures and thus the more reliable the information that is reported to the sink. This concept is illustrated in Figure 1.4.

Barrier Coverage

Barrier coverage refers to determining whether or not an event may pass across a path, or barrier, without being detected. This type of coverage in depicted in Figure 1.3 (c). Various types of barrier-based coverage have been proposed. Here, two such examples are presented. In [18], the authors combine an algorithmic approach with computational geometry for analysis of coverage in sensor networks. The key ideas developed are the concepts of worst-case and best-case coverage. The objectives of worst-case coverage are to find areas of "lower observability" from sensor nodes and to detect breach regions. Best-case coverage focuses on finding areas of "high observability" and identifying regions of best support.

The authors define worst-case coverage as the maximal breach path (MBP), or the path in which the distance between any point on the path and the closest senor node is maximized. This scenario is illustrated in Figure 1.6.

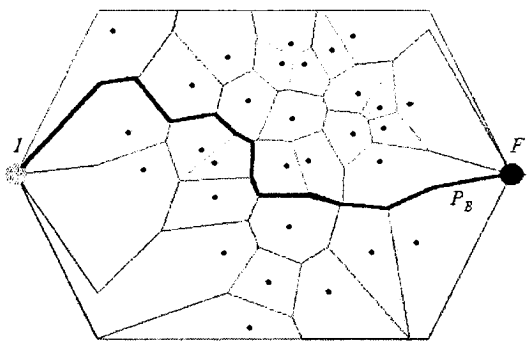


Figure 1.6: Sensor Field With Voronoi Diagram and Maximal Breach Path (MBP). [18]

This concept exploits the idea that sensing accuracy generally decreases as distance from a sensor node increases (i.e., the Probabilistic sensor model).

The maximal support path (MSP), or best-case coverage, is the case where the distance between any point on the path and the closest sensor node is minimized. The MSP is illustrated in Figure 1.7. The concepts of best and worst-case coverage highlight weak (i.e., MBP) and strong (i.e., MSP) points in senor fields which can be used by algorithm

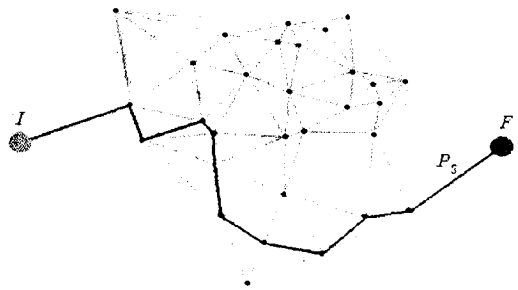


Figure 1.7: Sensor Field With Delaunay Triangulation and Maximal Support Path (MSP). [18]

and protocol designers for future node placement or node relocation.

Another model for barrier coverage is the exposure-based model. This concept was introduced in [19] and is built upon in [1]. This model also assumes that sensing accuracy diminishes with increased distance, but additionally accounts for the amount of time a senor is exposed to an event. Depending on the type of sensor, an integral or derivative model may be used to determine the intensity of the exposure. This intensity value is then used to develop detection models. If the intensity over a given time interval exceeds some threshold, then the event is detected, otherwise it is not. Analysis of coverage proceeds in a similar fashion to [18] in that it focuses on determining best-case and worst-case coverage scenarios. Best-case coverage is the path of maximal exposure and worst-case coverage is the path of minimal exposure.

Many of the previously mentioned formulations do well at thoroughly defining coverage for either a specific type of sensor or a specific application; however, there has not been much focus placed on accounting for variations in types of sensors and network operation. Part of the contribution of this work is to provide a general framework for coverage analysis of a network of Pan/Tilt/Zoom cameras which takes into account variations in the factors that affect network coverage and condenses camera coverage into metrics that can be used for comparison of networks of cameras with various parameter settings and various modes of network operation.

1.1.4 Sensors with Directional Sensing

Most of the previously mentioned coverage formulations assume omni-directional sensing capabilities; however, there are many types of sensors that are limited to directional sensing. For example, consider sonar and camera sensors where sensing is limited to an angle or field of view. In this case it is important to consider orientation of the sensors when designing algorithms and placement strategies. Figure 1.8 illustrates one example of a physics-based directional model of a sonar sensor.

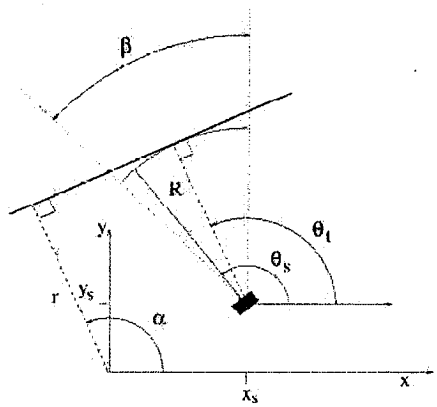


Figure 1.8: Plane target sonar sensor model. A plane is represented by the perpendicular distance r and orientation α. The shaded rectangle indicates a single sonar sensor located at the position (x_s, y_s, θ_s). [20]

A major contribution of this work will be development of a directional sensing model for the analysis of coverage when using cameras for sensing. Typically, related works have modeled a sensor's sensing range as a disk, where the radius of the disk is the sensing range. This assumes that there is no direction associated with sensor's sensing capabilities. In the case of a camera, sensing range is limited by the field of view and therefore has a direction associated with it. This work attempts to incorporate directional sensing into analysis of coverage of wireless sensor networks.

1.1.5 Time-varying (Dynamic) Coverage in Sensor Networks

While coverage of immobile sensors with omni-directional sensing capabilities has been extensively explored and is relatively well understood, only recently have researchers more extensively explored the idea of time-varying or dynamic coverage [5]. This type of network operation may be desirable in the case of having a limited number of sensor nodes, or when it is not required to cover every point of the area of interest simultaneously.

In the case of time-varying coverage, the sensing area of a sensor node changes over time. As a result, points that were originally uncovered may alternate between being covered and not being covered and thus a greater amount of area will be covered over time. This bring about the following questions to be answered [5]:

- What is the area coverage at a given time instant?
- What is the area coverage over a time interval?
- What are the durations of time that a location is covered and not covered?

Additionally, for static coverage of a sensor network, an intruder that is not initially detected will never be detected if it remains stationary or moves along an uncovered path. Detection of the intruder is more likely to occur if sensors can search an area. This concept brings about the following important questions to be answered by this work:

- How quickly can a network of PTZ cameras detect an intruder?
- How does the detection time depend on the searching strategy of the cameras?
- Can one derive camera-parameter adjustment strategies to improve intrusion detection capabilities?

In [5], the authors show that mobility can be used to offset a lack of sensors in order to improve network coverage. This work will attempt to use adjustment of camera instead of mobility to improve network coverage over time.

1.1.6 Camera Coverage Models

An additional contribution of this work will be the development of a model for coverage of a PTZ camera with respect to image processing application. Camera models are used to characterize the view of a camera using intrinsic and extrinsic camera parameters [17]. Intrinsic camera parameters specify internal camera characteristics and include focal length, principal point and effective pixel size. Extrinsic parameters describe the spatial relationship between the camera and the world. Exemplary extrinsic parameters are the rotation matrix and translation vector.

The inputs to a camera model are typically defined using camera parameters and the outputs represent the relationship between a point in the world reference frame and its image projection. One such example is the pinhole camera model [17]. Here, the relationship between a 3D point M and its image projection m is given by the formula:

$$m = A[Rt]M, \tag{1.2}$$

where A is the camera intrinsic matrix, R is the rotation matrix and t is the translation vector. The matrix A can be expressed as:

$$A = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}, \tag{1.3}$$

where (c_x, c_y) are coordinates of the principal point, (f_x, f_y) are the focal lengths by the axes x and y and the matrix R and vector t can be expressed as:

$$R = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}, \; t = \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix}. \tag{1.4}$$

Other camera models have been developed to determine the degree of uncertainty in measurements obtained from a camera sensor. In [11], the author develops a camera model to "gain a degree of confirmation or rejection" about a robot's current position which is

based solely on visual information. This is accomplished by extracting room features from the camera image and comparing them with features extracted from a pre-developed world model. Similar to the pinhole camera model, the model in [11] uses camera parameters as inputs, namely a camera image, the robot's hypothesis of its current position and the actual position of the robot. Unlike the pinhole camera model, however, the output of this model is the probability that the robot's hypothesis is correct. This can be thought of as a quality of service measurement.

For this work, the goal of the camera model is to characterize the coverage of a PTZ camera in the two-dimensional world using the focal length and pan parameters of the camera as inputs. The pan parameter will define the orientation of the camera relative to some reference frame, similar to the θ or α parameter illustrated in the sonar model in Figure 1.8. The focal length parameter sets the zoom level of the camera, thus a practical limit is imposed on the view range by the finite spatial resolution of the camera. Figure 1.9 illustrates a possible model for the field of view of a camera with a limited view range.

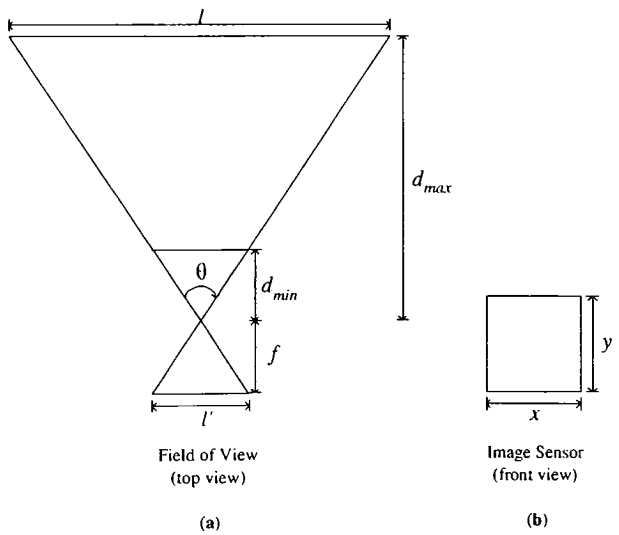


Figure 1.9: Camera Coverage Model

This work focuses on coverage in two different aspects, namely area coverage and intrusion detection. This raises the following questions to be answered:

- How can one model area coverage for a PTZ camera?
- What is the criteria for detection of an intruder (stationary or mobile)?
- How can one model detection time of an intruder?

A study of the related work reveals the following areas in which proposed camera models may be improved upon:

- **Unrealistic Assumptions**: Computational Geometry models, such as those presented in [24], make assumptions about the view of a camera that do not apply to real-world cameras, *e.g.*, infinite depth of field and infinite rotational speed. This work focuses on defining the real-world constraints for modeling camera coverage for a more practical and realizable approach.
- **Off-line vs. On-line Placement of Cameras**: Many active vision works have considered camera placement for *on-line* activity, but to our knowledge, little work has been done on the *off-line* placement of cameras prior to performing the active vision task.
- **Task-specific constraints**: Camera quality of service requirements vary for different tasks, thus there is a need for a generic camera model that incorporates the idea of task-specific parameters, such as the minimum acceptable spatial resolution.

1.1.7 Sensor Placement

For multi-camera vision systems, as the size of a network increases, it quickly becomes impractical for network designers to consider placing cameras by hand. Thus, it is important to consider algorithm for automated placement of cameras. Related works, such as [10] and [23], have addressed the problem of attempting to minimize the number of cameras needed for both manned and automated vision systems.

When considering real-world vision applications, it is important to consider the constraints that limit the coverage ability of a network of cameras. For example, [10] introduces the idea of constraining camera placement based on task-specific and floor plan-specific coverage requirements. In [23], the authors consider the placement of static cameras (sensors) in a dynamic scene.

A contribution of this work is to build upon related work and consider the placement of dynamic cameras in a static scene while also considering constraints specific to a particular floor plan or image processing application.

1.1.8 Visibility

Another important real-world consideration that needs to be taken into account when considering the coverage ability of multi-camera networks is constraints imposed by the presence of obstacles within the region of interest. This could include, for example, poles or walls within a building or an object moving in front of a camera and occluding its field of view.

Analyzing visibility becomes increasingly important when considering sensors that can vary their sensing area over time and scenes that change over time. [23] considers the effects on visibility of static cameras as a scene changes over time. This work considers the visibility of static and dynamic camera coverage models with respect to static scenes.

1.1.9 Covering Problems

Covering problems aim to find placements of a network of sensors so that, together, they cover a collection of target regions [9]. For example, [8] considers 2D covering polygons where the goal is to decide if a set of covering polygons can be translated to cover a target set. If the answer is"yes", then the polygons are rotated and placed. This is illustrated in Figure 1.10.

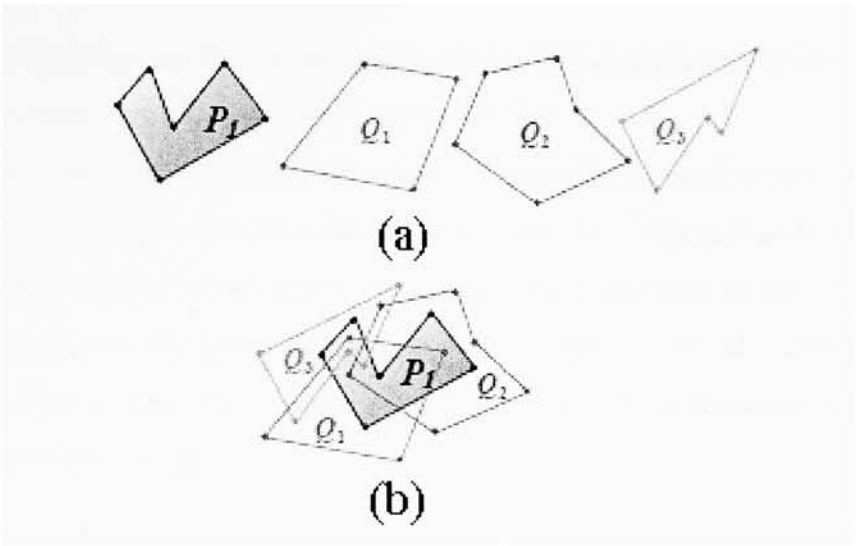


Figure 1.10: 2D Covering (a) Sample P and Q, (b) Translated Q Covers P

This work attempts to draw upon this concept by using cameras, represented as triangular regions, to cover a polygon representation of a floor plan. The polygon is decomposed into triangular regions and the question becomes, given camera and application constraints, can one place a camera and adjust its parameters such that the camera covers the triangle? If the answer is "yes", then the camera is placed and the parameters are adjusted.

Another important piece of information learned from the study of covering problems is that a number of 2D covering problems have been proved to be NP-complete [9]. This gives an idea of the complexity of the problem being dealt with and is additional support for a heuristic approach.

1.1.10 Camera Placement

Other related works have studied the problem of camera placement and have proposed solutions. [10] considers a more realistic model for a camera than those proposed in Computational Geometry works such as [24]. In addition, the authors in [10] propose methods for incorporating task-specific constraints into a model for camera coverage. The problem is posed as a discrete optimization problem where the goal is to minimize the cost of placing cameras to cover a room subject to task-specific constraints. Some interesting concepts

developed in [10] that will be drawn upon in this thesis include the formulation of a model for dynamic camera coverage and concept of visibility.

The authors present a method of developing a model for dynamic camera coverage that involves the introduction of a task-specific time constraint. This represents the minimum amount of time required for an object to be within the visible field of view of at least one camera in the system. By determining the sweeping angle of view of a camera over time, the field of view of the camera can then be approximated by a triangular region. Figure 1.11 illustrates this concept.

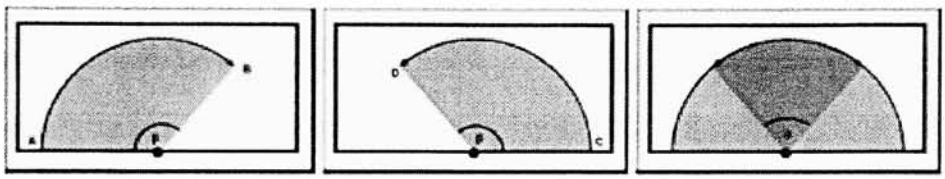


Figure 1.11: Illustration of the reachable region from a camera (black disk) location on the polygon perimeter.[10]

This work attempts to simplify this dynamic camera coverage model even further by only considering a partial range of motion instead of considering sweeping the camera over the entire range of motion.

Additionally, the authors in [10] present a method for dealing with occlusion of objects, or determining the "visibility" of a camera. To do this, an angle sweep technique is presented. While this work heavily leverages this concept, the implementation is vastly different in that [10] works in the discrete space, while this work attempts to work in the continuous space. Figure 1.12 illustrates the concept of polygon visibility over a discrete problem space.

1.2 Thesis Overview

This thesis presents an approach to the camera placement problem that involves determining a reasonable, yet simple model for camera coverage, developing and algorithm to place

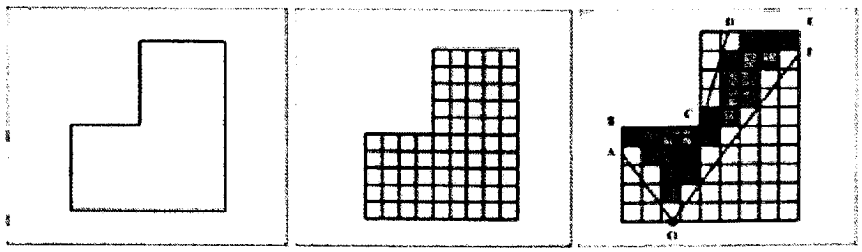


Figure 1.12: **Left**:The polygon. **Middle**:Cellular representation of the polygon. **Right**:The cell coverage of a camera O with FoV limits OA and OF and visible polygon OABCDEF. The dark cells are the visible ones from camera O.[10]

different shapes of triangles (representing the field of view of the cameras) that cover a 2D room and analyzing the camera placement algorithm via simulations to gain insights as to where to place cameras in order to improve coverage.

Chapter 2

Dynamic PTZ Camera Coverage Model

In order to make the problem more manageable, the constraints of the problem can be relaxed to only consider two-dimensions. Thus, the tilt aspect of a PTZ camera is ignored.

Based on the study of the related work and this simplification, it seems that a reasonable, yet simple enough approach is to model the dynamic coverage of each PTZ camera as a triangle that represents the field of view of the camera, and place different shapes and sizes of triangles to cover a 2-D room.

The approach is to first develop a model for static camera coverage, and then extend the model to incorporate dynamic coverage constraints. The goal is to represent the sweeping field of view of a camera in the shape of a triangle. This then gives us the ability to analyze a given camera placement algorithm, in terms of the efficiency, robustness, and practicality of the coverage provided by the network of PTZ cameras.

This section outlines proposed model for dynamic camera coverage. Here, consider the 2D area that can be covered by a camera with respect to a particular image processing application. For completeness, some important terms and camera parameters are outlined.

2.1 Pan/Tilt/Zoom Cameras

Various types of cameras are available for video surveillance tasks. This work focuses on the use of digital cameras with Pan/Tilt/Zoom (PTZ) controls.

- **Digital camera**: A *digital camera* uses an electronic sensor to acquire spatial variations in light intensity and then applies image processing algorithms to the data provided by the sensor to reconstruct a picture of a scene [4]. Two technologies currently used to manufacture digital image sensors are Charge Coupled Device (CCD) and Complementary Metal Oxide Semiconductor (CMOS). Both CCD and CMOS perform the task of converting light energy into electric charge to capture information about a scene. Image sensors can be thought of as a 2-D array of thousands or millions of tiny solar cells that each transform light from one small part of an image into electrons.

- **Pan/Tilt/Zoom (PTZ) camera**: A *Pan/Tilt/Zoom camera* is a special type of camera that has motorized control for rotating the camera in all directions and zooming in on an image.

2.2 Camera Parameters

Next, some important camera parameters are defined.

2.2.1 Format Size

An ideal lens produces images in the form of a circle, called the image circle. Digital cameras use a rectangular image sensor to capture a scene through a lens. Figure 2.1 illustrates the concept of *format size*.

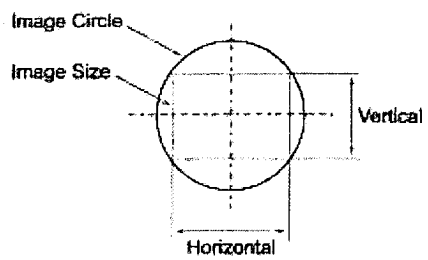


Figure 2.1: Format Size [16]

The ratio of horizontal length to vertical length of the image sensor is known as the aspect ratio of the camera. A typical aspect ratio is 4:3 (H:V) for a standard digital camera. Table 2.1 displays examples of common image circle sizes along with the corresponding horizontal and vertical lengths.

Image Circle	Horizontal	Vertical
⊘4.0mm	3.2mm	2.4mm
⊘6.0mm	4.8mm	3.6mm
⊘8.0mm	6.4mm	4.8mm
⊘11.0mm	8.8mm	6.6mm
⊘16.0mm	12.8mm	9.6mm

Table 2.1: Typical image circle sizes [16]

Lenses are designed to work with specific image sensor sizes. Different image sensor sizes will yield different fields of view using the same focal length. Common image sensor sizes include 1/4", 1/3", 1/2", 2/3" and 1". Figure 2.2 illustrates some typical image sensor sizes and the corresponding horizontal, vertical and diagonal lengths.

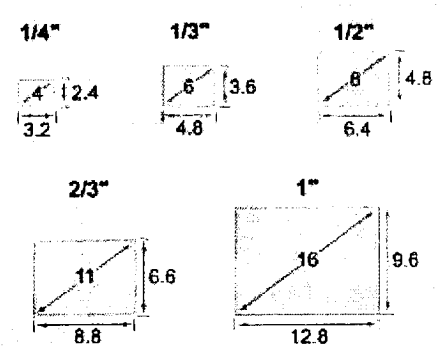


Figure 2.2: Typical image sensor sizes (units in mm). [16]

2.2.2 Effective Pixel Size

Most image acquisition units today utilize CCD technology. Compared to picture tubes, CCD devices have relatively lower production costs and have the advantage of being composed of many separate light-sensitive elements. Thus, CCD sensors inherently deliver a

two-dimensional, spatially discrete image. Figure 2.3 illustrates the 2-D array nature of a CCD sensor.

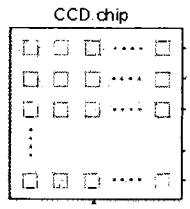


Figure 2.3: CCD sensor [29]

An important parameter of CCD chips is resolution, which is expressed as the number of pixel elements. A CCD chip typically has a few rows and columns of "blind" pixels at its edges, thus the area of the chip used for image acquisition is known as the *effective pixel size* of the chip. An example of a modern high resolution CCD chip is one that contains 756 pixels in the horizontal direction and 581 pixels in the vertical direction.

2.2.3 Focal Length

For a thin double convex lens, rays from an infinite light source will converge at a single point, called the *focal point*. The distance between the lens and the focal point is known as the *focal length* f. To optimally capture a scene at an infinite distance to an image sensor, the distance between the lens and the sensor is set to the focal length, *i.e.*, the sensor should lie on the *focal plane*. This is illustrated in Figure 2.4 (a).

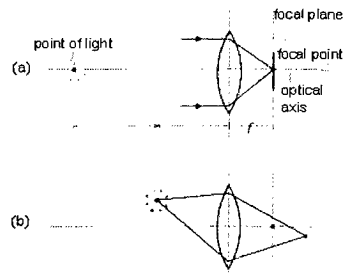


Figure 2.4: Focal length [30]

2.2.4 Angle of View

The *angle of view* corresponds to the amount of a given scene that appears on a sensor. Angle of view can be measured horizontally, vertically or diagonally. For digital cameras, angle of view is a function of focal length and the size of the image sensor [16]. This is illustrated in Figure 2.5.

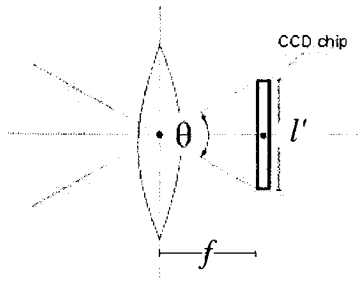


Figure 2.5: Angle of View

The formula for angle of view, using the ideal lens model, is represented in Equation 2.5.

$$\theta = 2\arctan(\frac{l'}{2f}), \tag{2.1}$$

Here, θ is the angle of view, f is the focal length of the camera, and l' is a dimension of the CCD chip (horizontal, vertical or diagonal). See Figure 2.2 for more details..

With a fixed image sensor size, there is an inverse, non-linear relationship between the focal length and the angle of view.

2.2.5 Field of View (FOV)

The volumetric region visible from a camera is defined as its *field of view* [10]. This is determined by the apex angles of the pyramidal region originating from the lens center of the camera, illustrated in Figure 2.6.

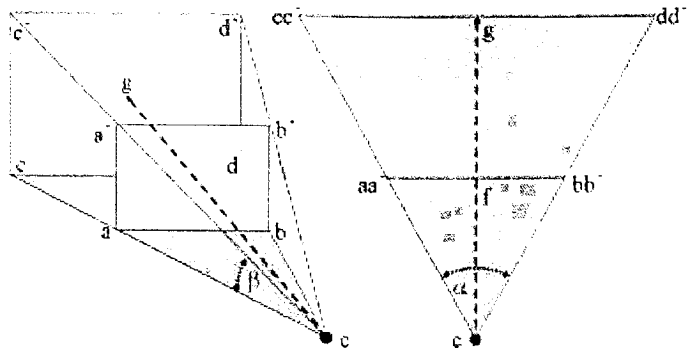


Figure 2.6: Field of View and Depth of Field. α and β are respectively azimuth and latitude of the Field of View. **c** is the camera, $\overline{\mathbf{cg}}$ is the optical axis, and the frustum defined by the planes **abb'a'** and **edd'e'** is the Depth of Field . [10]

2.2.6 Depth of Field (DOF)

The *depth of field* of a camera is the distance between the nearest and farthest objects that appear in acceptably sharp focus in an image [10]. This concept is illustrated in Figure 2.6 as the frustum defined by the planes **abde** and **a'b'd'e'**. Further decomposing *DOF* into two components:

- **Minimum Object Distance**: The distance between a camera and the nearest object that appears in acceptably sharp focus is defined as the *minimum object distance*. This is dependent upon the camera's focal length. The greater the focal length, the greater the *minimum object distance*.

- **Maximum Object Distance**: The distance between the camera and the farthest object that appears in acceptably sharp focus defines the *maximum object distance*. This parameter is dependent upon task-specific constraints, which may be spatial, temporal or based on quality of service. For example, consider the image processing application of human face detection. In this case, the *maximum object distance* would be limited by a minimum acceptable spatial resolution of face images required by face detection algorithms.

2.2.7 Spatial Resolution

Spatial Resolution is a measure of accuracy or detail of a digital image. As an example, for a digital image, this could mean the ratio between the total number of pixels needed to represent an object in an image and the size of the object in the real world. The higher the *spatial resolution*, the more detailed and sharper an image will be.

2.3 Application Parameters

Because this work considers the coverage of a camera with respect to an image-processing application, here some application-specific parameters are defined.

2.3.1 Object Size

This defines the size of the target object to be covered.

2.3.2 Required Pixels

This is an application-specific constraint that defines the minimum pixel resolution required for a particular image-processing algorithm.

2.4 Camera Coverage Parameters

For this work, the problem formulation is limited to two-dimensional regions in order to make the problem more manageable. The camera coverage model is defined in terms of inputs and outputs. The idea is to convert the camera parameters and task-specific constraints of the problem into a spatial coverage representation. In order to do this, some additional terms are defined.

2.4.1 Minimum Spatial Resolution

The minimum spatial resolution parameter, r_s, refers to an application-specific limit on the acceptable resolution of captured images. In this case, consider applications such as face detection and face recognition. Related works have presented heuristics for minimum acceptable resolutions for face detection and face recognition applications. For face detection, the minimum number of pixels needed in a face image is 20x20 [14]. The number of pixels required for face recognition applications is greater due to the demand for more detail. In this case, 120x120 pixels is suggested as a limit [25].

The minimum spatial resolution required for a given camera and application pair can be determined by dividing the required number of pixels by the size of the object.

2.4.2 Minimum Application Distance

The Minimum Application Distance, $d_{app,min}$, is defined as the minimum distance required for a target object to be completely contained within the field of view of the camera. Consider, for example, the application of face detection, it still holds true that there is optical blur for objects that are closer than the Minimum Object Distance to the camera; however, consider also the distances in which a target object is beyond the MOD but still may be close enough that the entire object does not fit within the field of view for the camera. Figure 2.7 illustrates the concept of Minimum Application distance.

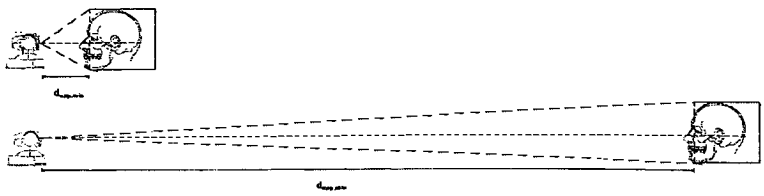


Figure 2.7: Minimum Application Distance (w.r.t. Face Detection/Recognition)

2.4.3 Maximum Application Distance

The Maximum Application Distance, $d_{app,max}$ is a function of focal length, effective pixel size and the minimum acceptable spatial resolution. This parameter is derived (referring to parameters in Figure 2.8) as follows:

From laws of similar triangles:

$$\frac{d_{max}}{f} = \frac{l}{l'}$$
$$d_{max} = f(\frac{l}{l'})$$

x defines the number of picture elements in the horizontal direction and l represents the width of an object that can be covered at distance d_{max}. So spatial resolution can be defined as $\frac{x}{l}$ in units of $\frac{px}{mm}$. And from the application, notice the spatial resolution needs to be greater than or equal to r_s:

$$\frac{x}{l} \geq r_s$$
$$l \leq \frac{x}{r_s}$$

Plugging in and solving:

$$d_{app,max} = f(\frac{\frac{x}{r_s}}{l'}) \tag{2.2}$$

2.5 Static PTZ Camera Coverage Model

With all the parameters defined, the inputs and outputs to the camera model can be defined.

Inputs:

- Format Size
- Effective Pixels
- Focal Length
- Object Size

- Required Pixels

Outputs:

- Angle of View
- Minimum Application Distance
- Maximum Application Distance

The output parameters define the area visible from the camera, illustrated in Figure 2.8 (a).

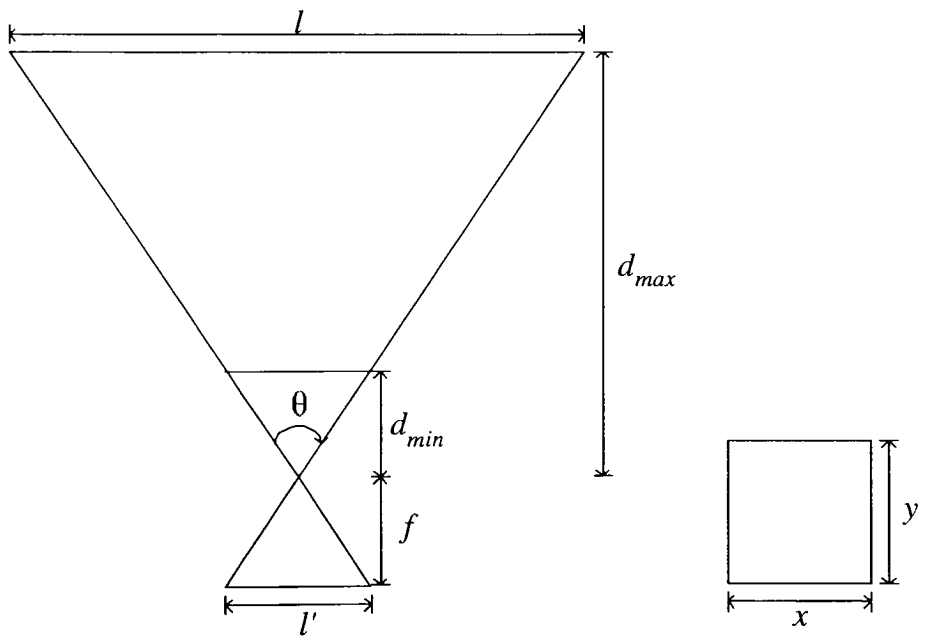


Figure 2.8: (a) Static camera coverage model and (b) image sensor parameters

2.6 Dynamic PTZ Camera Coverage Model

Now also consider the horizontal rotational ability of a camera when modeling camera coverage. If the camera is allowed to pan (rotate), one can see that there is potential to cover a much larger amount of area (as compared to the static coverage case) over a period of time. In doing this though, one must ensure that the target performance goals are not sacrificed.

This type of coverage analysis may be useful, for example, in applications that can tolerate areas being uncovered for a certain amount of time. Let T be an application-specific constraint that represents the maximum amount of allowable time that a point within a region can be uncovered. In other words, the goal is to ensure that for every point p within a region R, p will be guaranteed to be covered in a time less than or equal to T. Figure 2.9 is a graphical example of such a scenario.

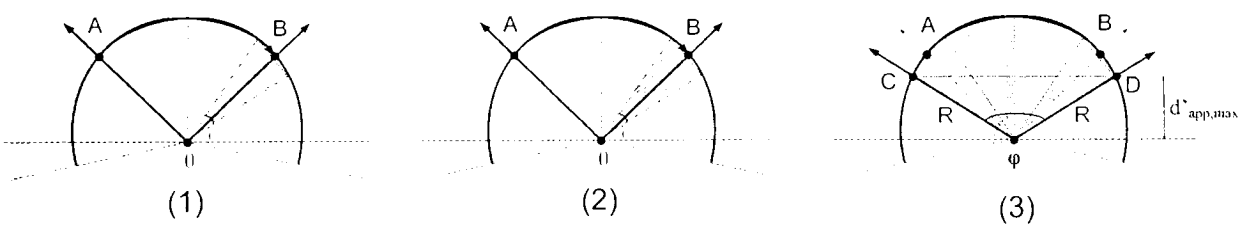


Figure 2.9: Dynamic Camera Coverage Model: (1) Static camera coverage with camera oriented toward point A, (2) Camera rotates ωT degrees to point in the direction of B and (3) Sweeping field of view of the camera (shaded region)

In Figure 2.9 (1), there is a camera at the origin oriented toward point A. Now consider that the camera can rotate, or pan, its field of view in the clockwise direction at horizontal angular velocity ω. If T is the application-specific time constraint, then the maximum angle that can be swept in that time is ωT. Consider the worst-case scenario where the camera needs to rotate an angle of ωT to cover a point. This is represented in Figure 2.9 (2) in which the camera is rotated from ωT degrees from orientation A to its new orientation B. The resulting angle that can be covered over time T is defined by Equation 2.3.

$$\phi = \theta + \omega T \tag{2.3}$$

Here, ϕ is the total effective horizontal angle of view over time T, θ represents the initial horizontal angle of view of the camera, and ω is the horizontal angular velocity. Note that if the rotational ability of the camera is not considered ($\omega = 0$), Equation 2.3 reduces to the original formula for horizontal angle of view (*i.e.*, $\phi = \theta$).

Figure 2.9 (3) displays the resultant angle, ϕ, after sweeping the camera from being oriented toward point A to point B. As such, this work will refer to ϕ as the sweeping angle of view from now on.

The sweeping angle of the camera creates a circular sector, defined as a wedge obtained by taking a portion of a disk with central angle $\theta < \pi$ radians (180 degrees). This shape, illustrated in Figure 2.10 has some well defined geometric properties. The radius R is equivalent to the maximum application distance, defined in Equation 2.2. The area of the circular sector is define in Equation 2.4, where A_s refers to the area of the sector. And the area of the triangular region of the circular sector is defined in Equation 2.5, where A_t refers to the area of the triangular region.

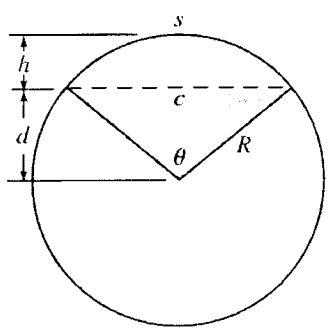


Figure 2.10: Circular Sector

$$A_s = \frac{1}{2}R^2\theta \tag{2.4}$$

$$A_t = \frac{1}{2}R\sin(\theta) \tag{2.5}$$

In order to simplify the analysis of dynamic camera coverage, it is possible to conservatively use the triangular region of the circular sector as the sweeping field of view. This gives us the ability to model dynamic coverage in the same manner as static coverage. Figure 2.11 illustrates the sweeping field of view concept.

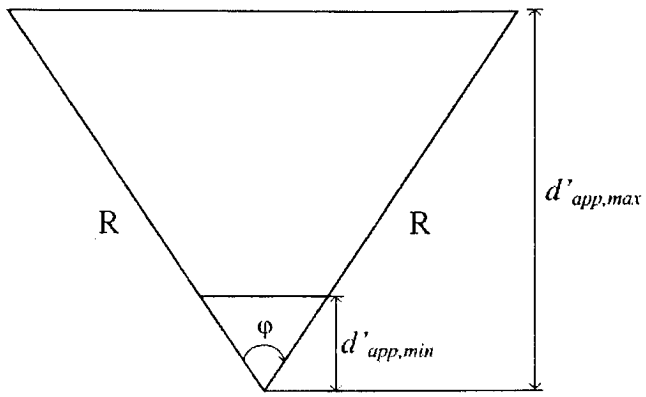


Figure 2.11: Sweeping FOV

The parameters from the model for static camera coverage (illustrated in Figure 2.8) are used to define the model for dynamic camera coverage. The model for dynamic camera coverage is summarized by Equations 2.3, 2.6, 2.7 and 2.8.

$$R = \frac{d_{app,max}}{\cos(\frac{\theta}{2})} \tag{2.6}$$

$$d'_{app,min} = d_{app,min} \tag{2.7}$$

$$d'_{app,max} = R\cos(\frac{\phi}{2}) \tag{2.8}$$

Having defined the size and shapes of room and objects of interest in this work and defining a model for dynamic camera coverage, the next step is to determine how the cameras may be placed within the room and how the visibility of a camera can be determined.

Chapter 3

Camera Placement and Visibility Algorithms

3.1 Procedure

The camera placement solution is a "divide-and-conquer" computational geometry approach. This approach attempts to take advantage of the triangular shape of the camera coverage model. The idea here is that a triangle is the most basic subcomponent of any polygon. So, if the polygon can be decomposed into triangular subcomponents, it follows that surely all of these subcomponents can be covered with a triangular region.

An important reason for choosing this approach is that by decomposing a polygon into triangular subcomponents, the algorithm guarantees that it will always provide complete coverage of the room (excluding the blind spot in the region around the camera due to the minimum application distance constraint). The basic procedure for breaking the room, represented as a simple polygon, into coverable components and placing cameras can be summarized as follows (see Figure 3.1).

1. Partition the room into more manageable sub-components (triangulation) and check for coverability of those sub-components
2. Further partition any remaining uncovered sub-components if necessary
3. Place the cameras and tune the camera parameters

4. Determine any overlapping visible camera regions (k-coverage)

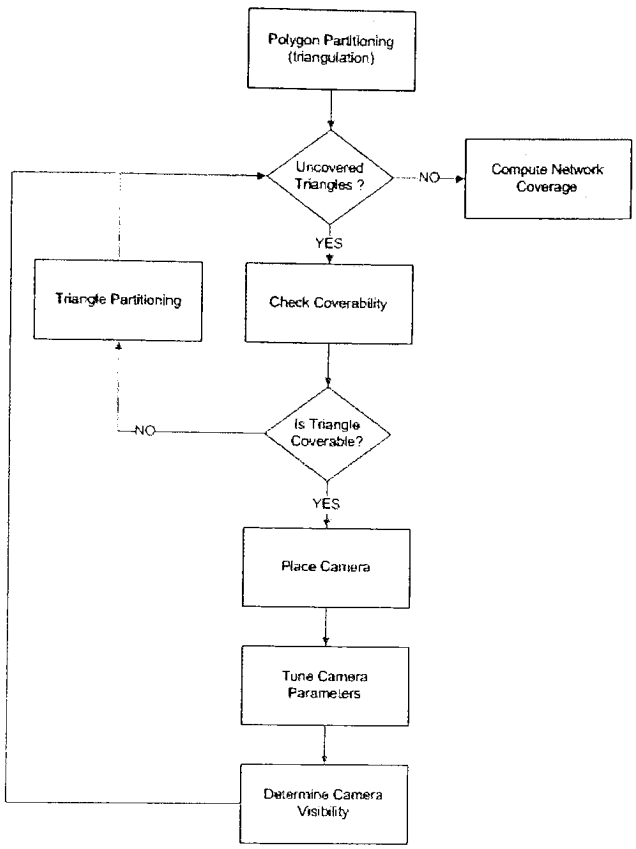


Figure 3.1: Procedure

First, the algorithm for camera placement is presented, followed by the method for determining the regions visible from a camera.

3.2 Camera Placement Algorithm

The approach of the camera placement algorithm is to decompose a polygon P into pieces that can be "covered" with one triangular camera field of view T. This concept is illustrated in Figure 3.2.

The input to the camera placement algorithm is a simple polygon representation of a building floor plan and the output is list of camera structures that provides coverage of the

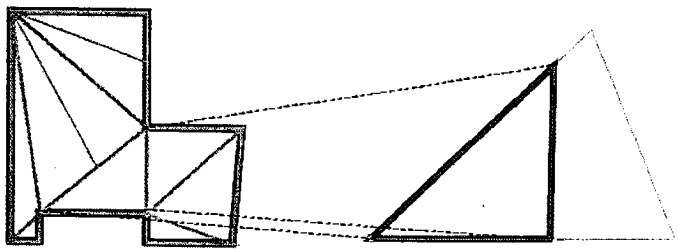


Figure 3.2: Camera Placement Algorithm

entire floor plan. The camera structures include the location, orientation and horizontal field of view for each of the cameras.

The camera placement algorithm assumes knowledge of the following:

1. Camera Specifications

2. Application Specifications

The following sections outline how the polygon is partitioned into more manageable sub-components and what it means to "cover" a triangular region.

The camera placement algorithm proceeds in three steps, outline in Algorithm 1.

Algorithm 1 *CameraPlacement()*

TriangulatePolygon()
EnqueueTriangles()
CoverTriangles()

Step 1: Triangulate Polygon

Polygon triangulation is a fundamental algorithm in computational geometry and has been found to be widely applicable in a number of fields, such as the tessellation of curves in computer graphics. As put by Seidel, this problem can be simply stated as "given the coordinates of the n vertices of a simple polygon P in order around P, find $n-3$ diagonals that partition P into $n-2$ triangles" [27].

The concept of polygon triangulation is illustrated in Figure 3.3.

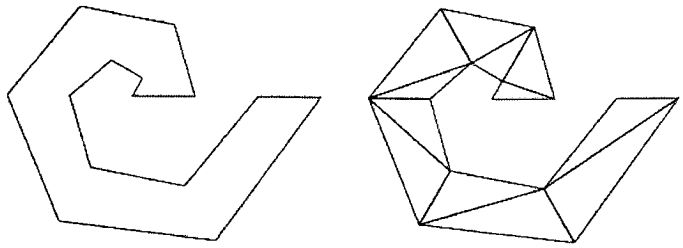


Figure 3.3: Polygon Triangulation [31]

This work uses a polygon triangulation presented by Seidel in [27]. This method was chosen because it is presented as being relatively simple and has low computational complexity. The algorithm for polygon triangulation proceeds as follows [31].

1. Decompose the Polygon into Trapezoids.

   Let S be a set of non-horizontal, non-intersecting line segments of the polygon. A randomized algorithm is used to create the trapezoidal decomposition of the X-Y plane arising due the segments of set S. This is done by taking a random ordering $s1..sN$ of the segments in S and adding one segment at a time to incrementally construct the trapezoids. This divides the polygon into trapezoids (which can degenerate into a triangle if any of the horizontal segments of the trapezoid is of zero length). The restriction that the segments be non-horizontal is necessary to limit the number of neighbors of any trapezoid. However, no generality is lost due to this assumption as it can be simulated using lexicographic ordering. That is, if two points have the same y-coordinate then the one with larger x-coordinate is considered higher. The number of trapezoids is linear in the number of segments. Seidel proves that if each permutation of $s1..sN$ is equally likely then trapezoid formation takes $O(nlog * n)$ expected time.

2. Decompose the Trapezoids into Monotone Polygons.

   A monotone polygon is a polygon whose boundary consists of two y-monotone chains. These polygons are computed from the trapezoidal decomposition by checking whether the two vertices of the original polygon lie on the same side. This is a

linear time operation.

3. Triangulate the Monotone Polygons.

   A monotone polygon can be triangulated in linear time by using a simple greedy algorithm which repeatedly cuts off the convex corners of the polygon. Hence, all the monotone polygons can be triangulated in O(n) time.

Step 2: Enqueue Triangles

Each of the triangles created by the triangulation algorithm are then added to a working queue. This maintains a running a list of regions that may not be covered by a camera.

Step 3: Cover Triangles

After the polygon is partitioned and enqueued in the working queue, the algorithm then continually attempts to pull from the working queue and "cover" the triangular region with a camera, represented in terms of the defined model for dynamic camera coverage. The criteria for coverage can be summarized as follows.

1. Minimum angle in the triangle $\leq$ Maximum camera angle of view

2. Longest edge next to Minimum angle of triangle leq longest camera depth of field

The concept of "covering" a triangular region with a camera is illustrated in Figure 3.4.

If successful in "covering" a triangle with a camera, it is removed from the queue and the camera is added to the list of place cameras. Otherwise, the algorithm attempts to partition the triangle into smaller triangles and then add these triangles to the queue. By continually attempting to cover smaller and smaller triangles, the algorithm will eventually converge into regions that can be covered.

The details for this step are laid out in Algorithms 2, 3 and 4.

3.3 Camera Visibility Algorithm

It is important to note that the camera placement algorithm only guarantees that each particular triangle that it produces is covered by at least one camera. It does not take into

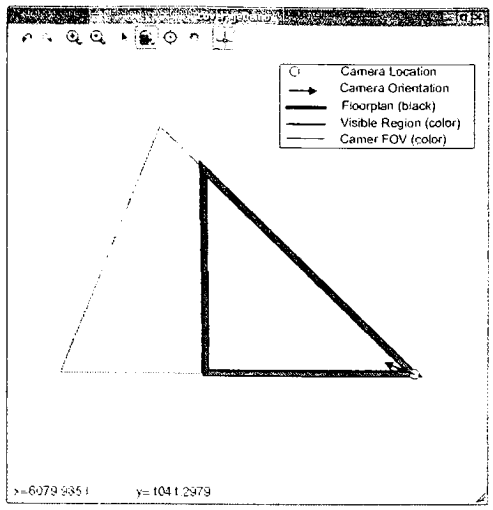


Figure 3.4: Camera Coverage

Algorithm 2 *CoverTriangles()*

```
while NOT TriQ.empty() do
    Dequeue first triangle
    Set parameters of camera C to cover triangle T
    CoverTriangle(T,C) // Can triangle T be covered by camera C
    if isTriCoverable then
        PlaceCamera()
    else
        PartitionTriangle()
        EnqueuePartitions()
    end if
end while
```

consideration the floor plan as a whole and does not take into account the additional coverage benefits provided by camera fields of view that extend past the particular triangular region that they are covering. In this case, the walls of the room may occlude the view of the camera.

In order to get the complete picture of the coverage provided by the set of cameras decided by the camera placement algorithm, it is first necessary to determine the area of the room that is visible from each camera.

The inputs to the visibility algorithm are a simple polygon representation of a building floor-plan P and a triangular region T representing the field of view of a camera. The

Algorithm 3 $CoverTriangle(T, C)$

```
i ⇐ T.getMinVertexAngleIndex()
isTriCoverable ⇐ FALSE

if T.getVertexAngle(i) leq ANGLE_OF_VIEW_H_MAX then
    orientation = ComputeOrientation(T, i)
    camera ⇐ Camera(T[i], orientation, T.getVertexAngle(i))
    a ⇐ left vertex of camera
    b ⇐ center vertex of camera
    c ⇐ right vertex of camera

    if Length(b, a) ≥ Length(T[i], T[i].left()) AND
      Length(b, c) ≥ Length(T[i], T[i].right()) then
        isTriCoverable ⇐ TRUE
        break
    end if
end if

return isTriCoverable
```

Algorithm 4 $PartitionTriangle(T)$

```
p ⇐ 1st vertex of T
q ⇐ 2nd vertex of T
r ⇐ 3rd vertex of T

part[0] = Triangle(p, angleBisector(p, q), angleBisector(p, r))
part[1] = Triangle(angleBisector(p, q), q, angleBisector(q, r))

return part
```

output is visibility polygon V which represents all of the points visible from the camera. The visibility algorithm is outlined in Algorithms 5 and 6.

3.3.1 Ray Shooting

The approach to camera visibility analysis is a simple ray shooting technique. Following the camera placement algorithm, in which the location, orientation and angle of view for a camera is determined, it is possible to compute the visibility by sweeping over the angle of view of the camera and shooting a ray in any given direction from the location of the camera. All of the points along the line segment defined by the camera location and the

Algorithm 5 *ComputeVisibilityPolygon*(P, T)

$I \leftarrow$ ComputeIntersectionPolygon()
$L \leftarrow$ ConstructVertexList()
$V \leftarrow$ ComputeVisibilityPoints()
SortVisibilityPoints(V)

Algorithm 6 *ComputeVisibilityPoints*(L)

$E \leftarrow$ ComputerEventPoints(L)
$V \leftarrow$ HandleEventPoints(E)

intersection point of the ray with the first of either a wall (if the view is occluded) or the limits of the camera field of view (defined by the maximum application distance) are considered to be "visible" from the camera. The visibility problem is illustrated in Figure 3.5 in which there is a room represented as polygon $P = (a, b, c, d, e, f)$ and camera represented as triangle $T = (g, h, i)$.

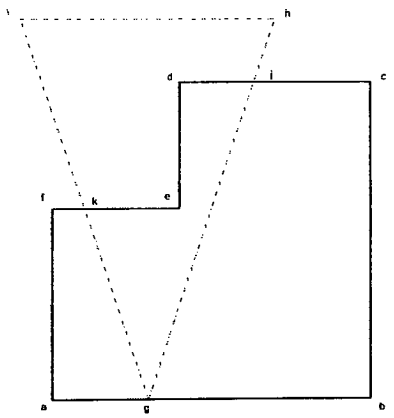


Figure 3.5: Camera Visibility Example

3.3.2 Polygon Intersection

As an initial level of filtering, first clip off any regions of the camera field of view that are outside of the polygon of interest. This can be accomplished by determining the intersection polygon I of polygon P and triangle T. This concept is illustrated in Figure 3.6.

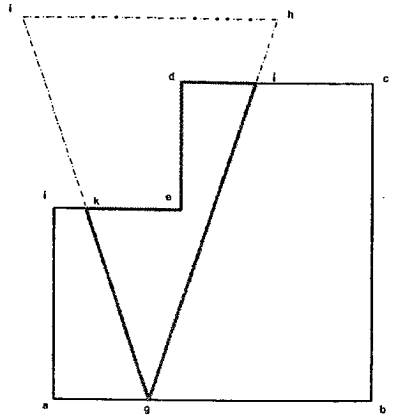


Figure 3.6: Camera Visibility Algorithm: Intersection

The resulting intersection is polygon $I = (g, j, d, e, k,)$.

Note that it is possible for the intersection of P and T to produce multiple polygons that are not contiguous. In this case, the polygon that contains the point representing the location of the camera is the only region of importance.

3.3.3 Event Points

An interesting observation that further simplifies the visibility analysis is that the visibility of a ray emanating from a camera location will only ever be affected at the vertices of the room. Thus, it is sufficient to only consider these points for the ray shooting technique. This corresponds to the ConstructVertexList() function in 5.

Computing Event Points

For each ray R, the algorithm determines the intersection with any wall of the intersection polygon I. Each of these points is stored in a list. These points are termed *event points*. These are the points that must be considered to determine whether or not a particular vertex alters the FOV of the camera, i.e., some "event" takes place at these points.

Note that there are two possible cases of visibility events that can occur. These cases are illustrated in Figure 3.7:

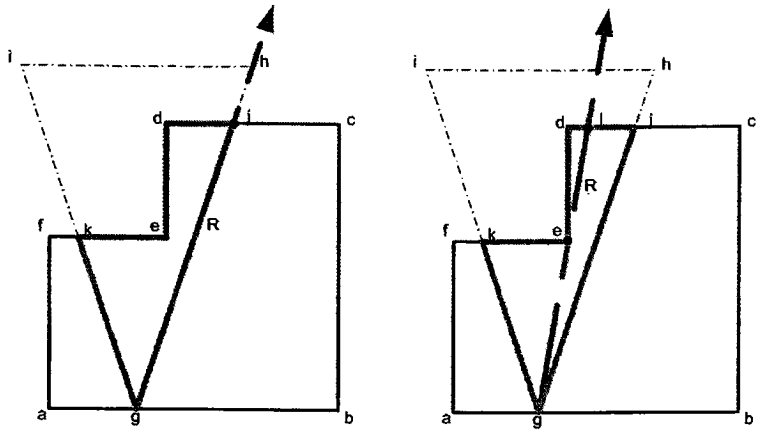


Figure 3.7: Camera Visibility Algorithm: Event Points

In case I, there is only one intersection point between ray R and the segments of the intersection polygon I. Here, the event point is simply added to the list of visibility points.

(Note that the segments that are collinear to ray R are considered to be visible, thus one is only concerned with finding the intersections with non-collinear segments)

Case II is a situation in which the ray may pass through one or more vertices before intersecting a segment that either occludes the field of view or defines the limits of the range of the FOV. Here, the algorithm marks all of these points as event points, as they will define the visibility polygon.

Handling Event Points

Because there is no guarantee that the event points will be collected in counter-clockwise order, the event points must be ordered after being collected. The counter-clockwise ordering of these points defines the visibility polygon V.

To achieve ordering, one can use the information from the intersection polygon. Traverse the edges of the intersection polygon in CCW order and add each event point that is on that particular segment to a list of intersection points. Since each segment of the intersection polygon is defined in CCW order, there is a source point that defines the beginning

of the segment, and a target point which defined the end of a segment. The list of intersection points can be sorted by distance from the source point to maintain ordering. After sorting, these intersection points are then added to the list of visibility points.

Figure 3.8 shows the result of the visibility algorithm for the example camera and room.

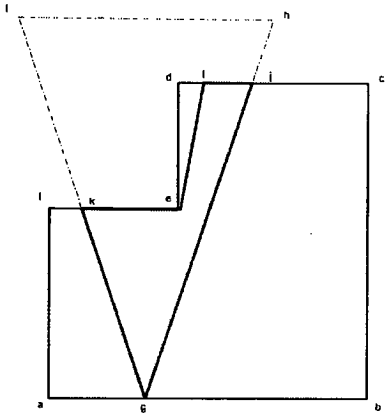


Figure 3.8: Camera Visibility Algorithm: Visibility Polygon

The visibility polygon is polygon (g, j, l, e, k).

This chapter has defined the algorithms for determining the placement and visibility of cameras. This leads into the discussion of how these algorithms are implemented in a simulated environment to gain insight into the problem and determine the performance of the camera placement algorithm

Chapter 4

Simulated Environment and Analysis of Dynamic PTZ Camera Coverage

A simulation framework was developed (for which a visibility algorithm was needed) to allow for the identification of needed improvements for the camera placement algorithm and to gain insight on strategies for camera placement and camera parameter adjustment to improve network coverage.

4.1 Simulated Environment

The simulated environment computes the placement and parameter adjustments of a network of PTZ cameras given the specifications for (1) an image processing application, (2) a camera and (3) a building floor plan. It also computes the resulting coverage and outputs a coverage report and plots of the camera placement and level of coverage (k-coverage).

4.1.1 Application Specifications

Information fed into the simulated environment about the target application includes specifications for the size of the object, the pixel requirements for any image processing algorithms and an application specific time constraint specifying the system's tolerance for the minimum amount of time required for a target to become visible by at least one camera

within the network.

For example, consider face detection as the target image processing application. Then the application specifications for object size become the breadth and height of a human face, illustrated in Figure 4.1. From [32], note that the average size of a human head for adult U.S. males is about 150mm in breadth and 240mm in height.

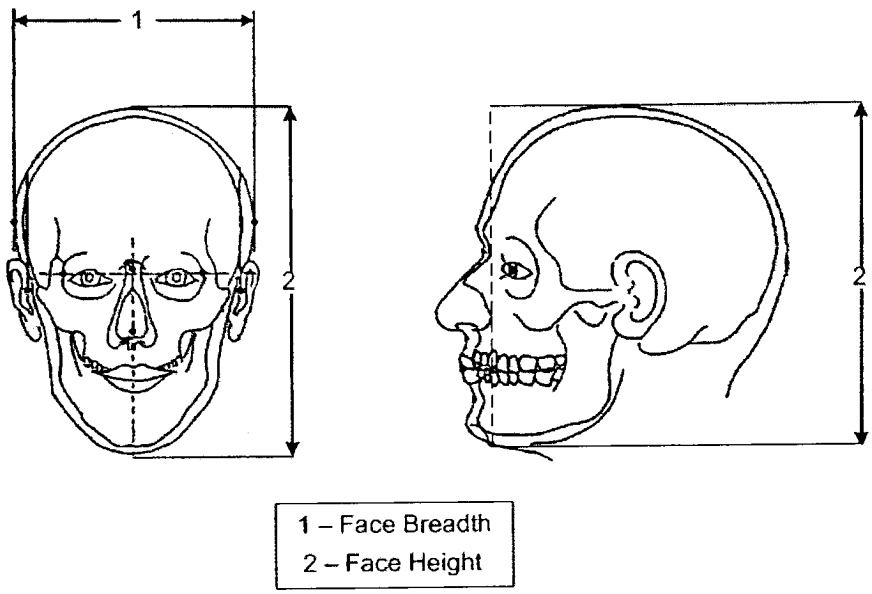


Figure 4.1: (1) Face breadth and (2) Face height [32]

For pixel requirements, [14] tells us that face detection algorithms require image resolutions of 20x20 pixels.

The time constraint may vary depending on the desired performance goals of the systems. Consider $T = 0.5$s as an example.

Table 4.1 summarizes the inputs to the simulated environment for the application of face detection.

4.1.2 Camera Specifications

Important camera specifications for the simulated environment include the format size of the camera, number of effective pixels, the range of focal lengths and the range of horizontal

Parameter	Value
Object Size (H)	150.0 mm
Object Size (V)	240.0 mm
Required Pixels (H)	20 px
Required Pixels (V)	20 px
Time Constant	0.5 s

Table 4.1: Face Detection Specifications

angular velocity. For this work, consider the use of the Sony EVI-D100 camera as an example. Figures 4.2 and 4.3 summarize the camera specifications for the Sony EVI-D100 camera.

	EVI-D100	EVI-D100P
Video Signal	NTSC	PAL
Image Sensor	1/4 type Super HAD CCD™	
Effective Pixels	768 (H) x 494 (V)	752 (H) x 582 (V)
Horizontal Resolution	470 TV lines (Wide end)	460 TV lines (Wide end)
Lens	10x Optical Zoom, 40x with Digital Zoom, f = 3.1 to 31, F 1.8 to 2.9	
Horizontal Angle of View	6.6 to 65 degrees	
Minimum Object Distance	100 mm (Wide end), 600 mm (Tele end)	
Minimum Illumination	3.5 lx (F1.8)	
Auto Exposure	Auto/Manual/Priority AE, Exposure compensation, Back light Compensation	
Shutter Speed	1/4 to 1/10,000 s	1/3 to 1/10,000 s
White Balance	Auto, ATW, Manual, One Push, 3200K, 5800K	
Effects	Mirror Image, Still Image on Field Memory, Pastel Image, Negative/Positive Reversal, Sepia Image, Monochrome Image, Enhanced Contrast, Mosaic Image, Vertical Stretch, Horizontal Stretch, Motion Image on Still Image, Continuous Still Image, Motion Images on Binaried Still Image, After-image Lag of Moving Object	
S/N Ratio	More than 50 dB	
Pan/Tilt	Horizontal ± 100 degrees (Max speed 300 degrees/s), Vertical ± 25 degrees (Max speed 125 degrees/s) (in 0.07 degrees increments)	
Video Output	VBS, Y/C	
Power Requirements	DC 10.8 to 13.0 V	
Power Consumption	Max. 13.2 W (at 12 Vdc)	
Operating Temperature	0 to 40°C (32 to 40°F)	
Storage Temperature	-20 to 60°C (-4 to 140°F)	
Dimensions	4 1/2 x 4 3/4 x 5 1/4 inches (113 (W) x 120 (H) x 132 (D) mm)	
Mass	1 lb 14 oz (860 g)	
Supplied Accessories	AC adaptor, IR Remote Commander unit, Velcro tape, Operating instruction	

Figure 4.2: Sony EVI-D100 Camera Specifications [28]

4.1.3 Floor plan

A typical building floor plan is represented as a collection of planar regions, thus it makes sense to use a 2-D formulation to determine the placement of cameras in the floor plan. These regions are approximated by polygons to further simplify the problem. Figure 4.4

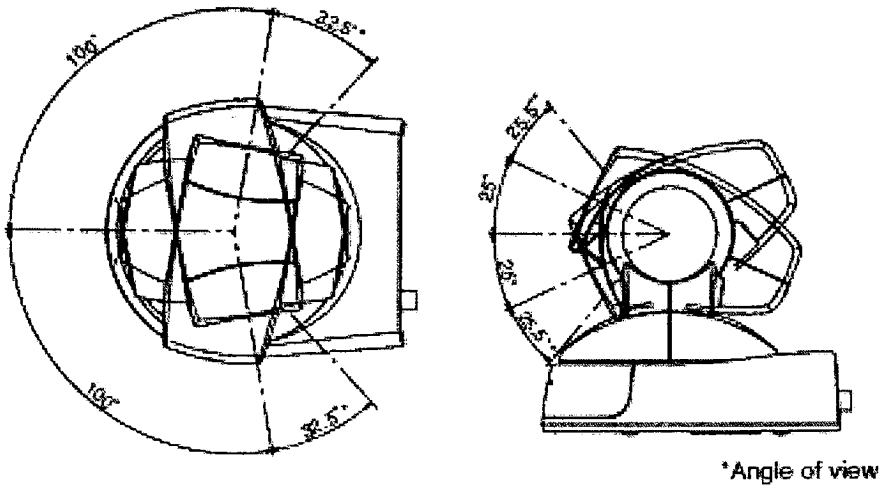


Figure 4.3: Sony EVI-D100 Pan/Tilt Range [28]

illustrates a typical building floor plan and how regions can be approximated by simple polygons.

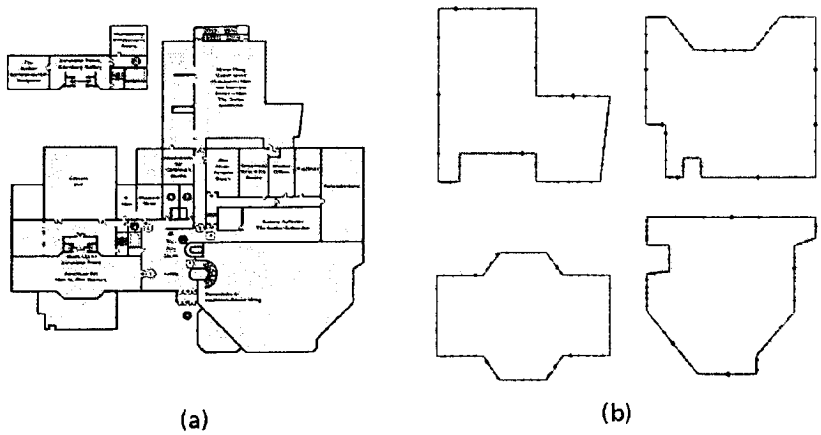


Figure 4.4: (a) A typical floor plan and (b) its polygon approximations. [10]

4.1.4 Coverage Metrics

In order to measure the performance of the camera placement algorithm, metrics were defined to analyze coverage in terms of efficiency, practicality and robustness.

Efficiency

Efficiency refers to maximizing the use of the cameras area coverage capabilities, *i.e.,* minimizing the number of cameras needed to cover a room. Efficiency is measured in simulations in terms of the number of cameras, the amount of area covered and uncovered per camera and the percentage of area covered and uncovered per camera.

Practicality

Practicality refers to the fact that certain camera placements are more desirable in real-world applications than others. For instance, it does not make much sense, practically speaking, to place multiple cameras in the same location as it offers a single point of failure for an object to occlude the view of the cameras. In simulations, practicality is measured in terms of the percentage of unique camera locations.

Robustness

Robustness refers to a networks ability to reliably perform a particular task despite failures or other outside interference. For example, for camera networks, this could mean that if one camera's field of view is occluded, the task can still be performed because the are other cameras with overlapping coverage areas. This is measured in the simulation environment in terms of k-coverage, or the number of overlapping sensors at any point within the network.

4.1.5 Area Coverage Analysis

In order to get an approximate idea of the number of cameras needed to cover a room of a given size, consider plugging in the application and camera specifications mentioned above into the models for static and dynamic camera coverage.

For example, by plugging in the camera specifications into the equation for angle of view, Equation 2.1, one can determine the relationship between focal length and angle of

view. For the Sony EVI-D100 camera, the image sensor is 1/4 in. type which corresponds to a horizontal distance of 3.2mm. Figure 4.5 shows the corresponding plot of angle of view (H) vs. focal length for the Sony EVI-D100 camera.

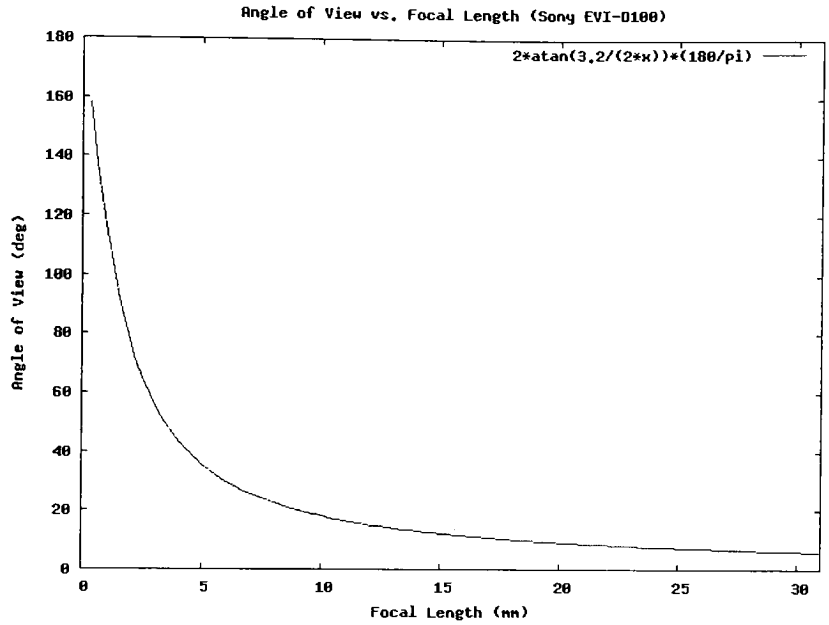


Figure 4.5: Angle of View (H) vs. Focal Length (Sony EVI-D100)

This plot can be used to derive the minimum and maximum horizontal angle of view of the camera. This is summarized in Table 4.2.

Parameter	Spec Value	Model Value	% Error
Angle of View (H) Min.	6.6 deg	5.9 deg	10.6
Angle of View (H) Max.	65.0 deg	54.6 deg	19.0

Table 4.2: Horizontal Angle of View (Sony EVI-D100)

Note that the specifications for the Sony EVI-D100 camera list the minimum and maximum horizontal angle of view of the camera to be 6.6 degrees and 65 degrees respectively, which vary from the calculated values. The difference can be attributed to the fact that this work considers modeling the lens of the camera using the simple lens model, where in actuality, the lens of the camera may be more complex. Using the simple lens model,

however, gives us a close approximation that is useful for gaining an understanding of the problem at hand.

The formula for coverage area can be derived as follows:

Let $k_1 = l'$, $k_2 = \frac{\frac{x}{r_s}}{l'}$, then:

$$A = \frac{1}{2}b \times h = \frac{1}{2}(k_2 f) \times (k_2 f)\tan(\frac{\theta}{2})$$
$$A = \frac{1}{2}(k_2 f) \times (k_2 f)\tan(\frac{2\arctan\frac{k_1}{2f}}{2})$$
$$A = (k_2 f)^2 \times \tan(\arctan\frac{k_1}{2f})$$

$$A = \frac{k_1 k_2^2}{2} \times f \tag{4.1}$$

The resulting equation (Equation 4.1) shows that area coverage is a linear function of focal length. This is illustrated in Figure 4.6.

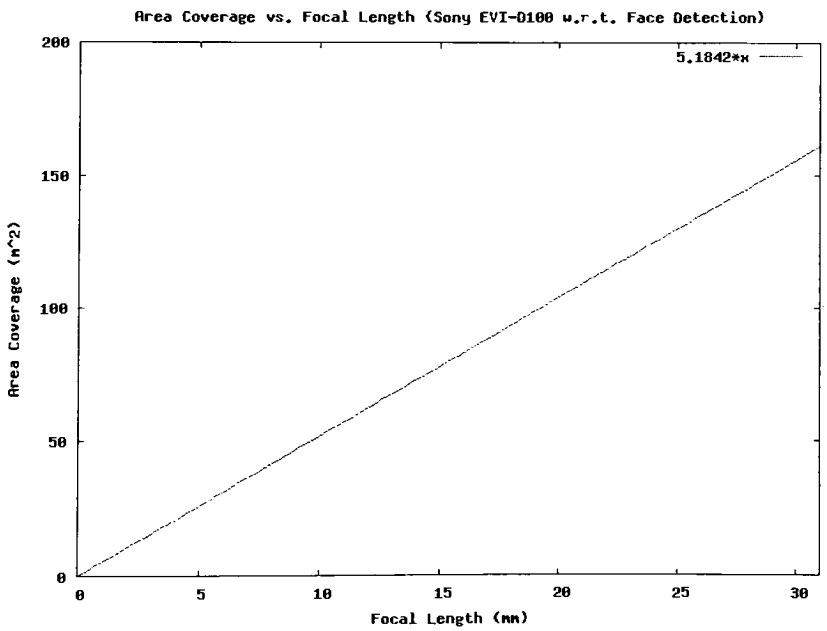


Figure 4.6: Coverage Area vs. Focal Length (Sony EVI-D100 w.r.t. Face Detection)

Extending this analysis to the dynamic model for camera coverage, To do this, consider the equation for the area of a circular sector and the area of the triangular region of a circular sector, represented by Equations 2.4 and 2.5, respectively. Figure 4.7 plots both of these

Focal Length	Coverage Area
3.1 mm	16.1 m^2
31.0 mm	160.1 m^2

Table 4.3: Focal Length and Corresponding Coverage Area (Sony EVI-D100 w.r.t Face Detection)

variables against the sweeping angle of view of the camera (from 0 to 180 degrees) for the unit circle (R=1).

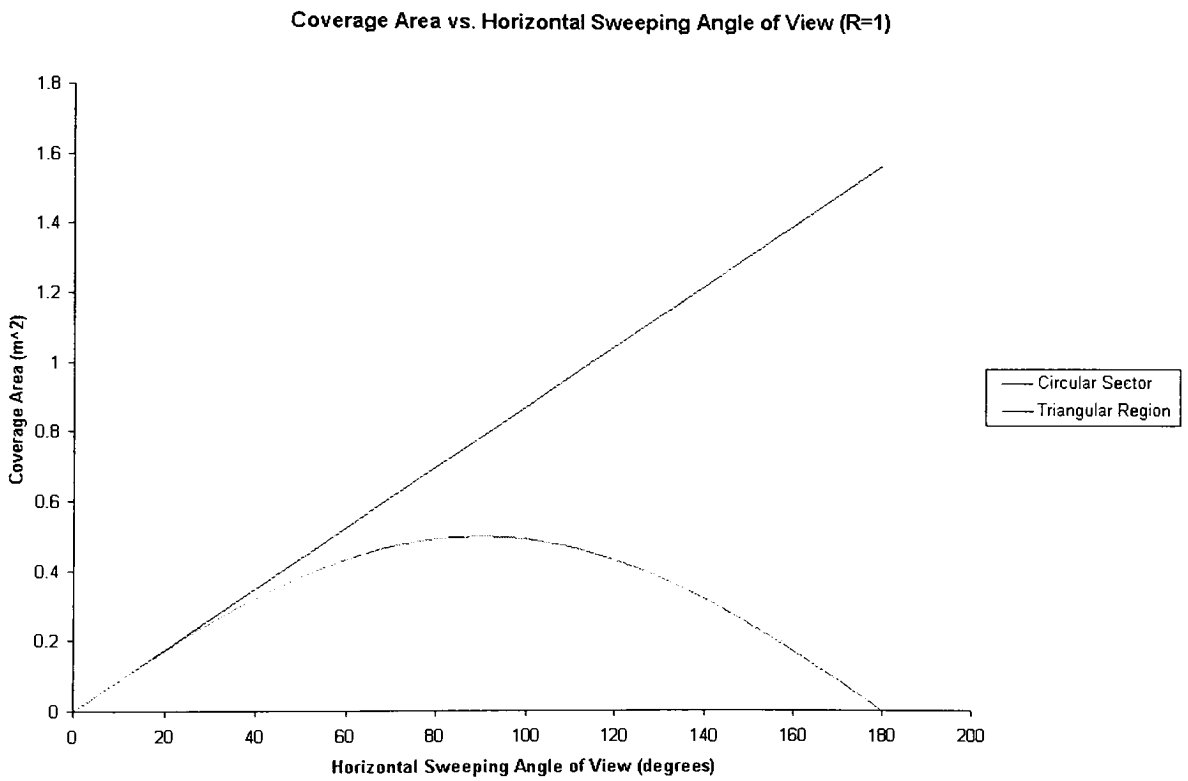


Figure 4.7: Circular sector area analysis (R=1)

Notice from Figure 4.7 that using the triangular region of the circular sector as an approximation of the coverage area is reasonably accurate for angles $\leq$ 90 degrees; however, above 90 degrees, there is a quick loss in accuracy. Table 4.4 summarizes the percent error associated with the approximation.

Note that the camera model is reasonable considering angles less than or equal to 90

Theta (deg)	Sector Area (m^2)	Area triangle (m^2)	Percent Error
0	0	0	0
10	0.09	0.09	0.25
20	0.17	0.17	1.35
30	0.26	0.25	3.38
40	0.35	0.32	6.34
50	0.44	0.38	10.18
60	0.52	0.43	14.83
70	0.61	0.47	20.2
80	0.7	0.49	26.19
90	0.79	0.5	32.7
100	0.87	0.49	39.61
110	0.96	0.47	46.8
120	1.05	0.43	54.14
130	1.13	0.38	61.51
140	1.22	0.32	68.78
150	1.31	0.25	75.85
160	1.4	0.17	82.59
170	1.48	0.09	88.92
180	1.57	0	-

Table 4.4: Circular Sector Area Analysis (R=1)

degrees. This work will only consider horizontal angles of view within this range. This can be accomplished by constraining the upper bound of the maximum horizontal angular velocity of the camera. For example, sticking with the case study, one can limit the maximum horizontal angular velocity by solving for ω in Equation 2.3. Figure 4.8 plots the coverage area for the dynamic coverage model versus the area for the circular sector using $\omega = 70\frac{deg}{sec}$.

This gives an idea of the amount of area that can be covered by considering the ability of the camera to sweep over time. Also, this gives us confidence in the accuracy of the camera model. Table 4.5 tabulates the range of areas for the various zoom levels of the camera (1x - 10x).

More complex models may consider the entire coverage area of the circular sector or perhaps multiple smaller fields of view placed next to one another, similar to area approximations used for integration.

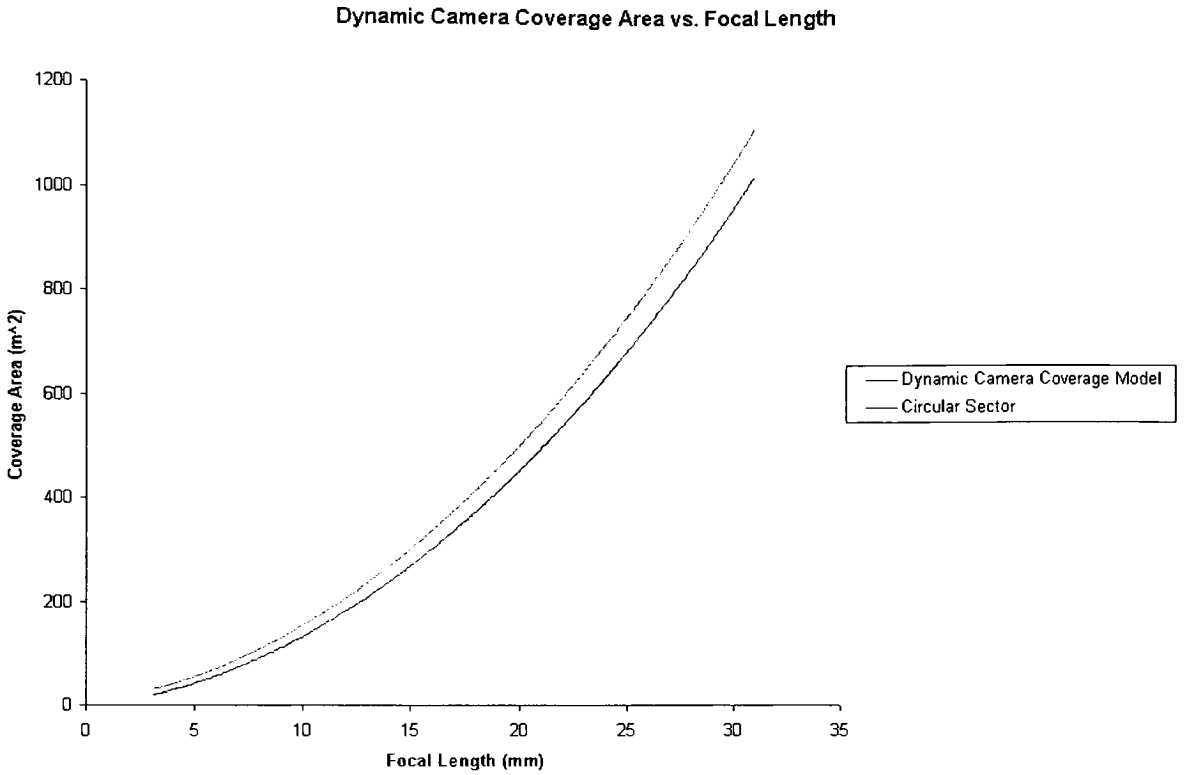


Figure 4.8: Dynamic camera coverage model area analysis

4.1.6 Implementation Details

The simulation framework was a custom environment developed using the C++ programming language with the use of the open source Computational Geometry Algorithms Library (CGAL). This code base provides efficient and proven implementations of computational geometry algorithms.

Other leveraged software libraries include an implementation of a fast and efficient algorithm for polygon triangulation developed by [22], which was used for the implementation of the camera placement algorithm, and an open source implementation of a polygon clipping algorithm, clippoly (see [26]), which was used in the implementation of the camera visibility algorithm for computing the intersection polygon.

Simulations were run on a Linux workstation running Gentoo Linux with a 2.6GHz

Focal Length (mm)	Computed Area (m^2)	Model Area (m^2)
3.1	19.7248	30.8464
6.2	59.6975	74.1601
9.3	117.5383	137.3497
12.4	193.2471	219.7972
15.5	286.8240	321.3674
18.6	398.2689	442.0143
21.7	527.5818	581.7178
24.8	674.7628	740.4678
27.9	839.8118	918.2588
31	1022.73	1115.0873

Table 4.5: Dynamic camera coverage model area analysis

Intel Pentium D processor and 1GB of RAM.

The following sections outline results obtained from the simulated environment.

4.2 Critical Variables for Camera Placement

The camera placement algorithm was first run on a series of rooms covering the various types of triangles, namely (1) acute, (2) obtuse, (3) right and (4) equiangular. These shapes are illustrated in Figure 4.9. The size of the rooms was varied from 50 square meters to 250 square meters in order to produce rooms requiring relatively large numbers of cameras.

The goal of testing on these basic shapes was to observe the behavior of the initial camera placement algorithm and determine the critical variables for camera placement and their effects on coverage.

4.2.1 Partitioning

It was found that the camera placement algorithm is highly dependent on the scheme used for decomposing a polygon into triangles. The initial camera placement algorithm used polygon triangulation for coarse partitioning and split the smallest angle of a triangle using an angle bisector scheme for fine partitioning; however, there are a number of other ways

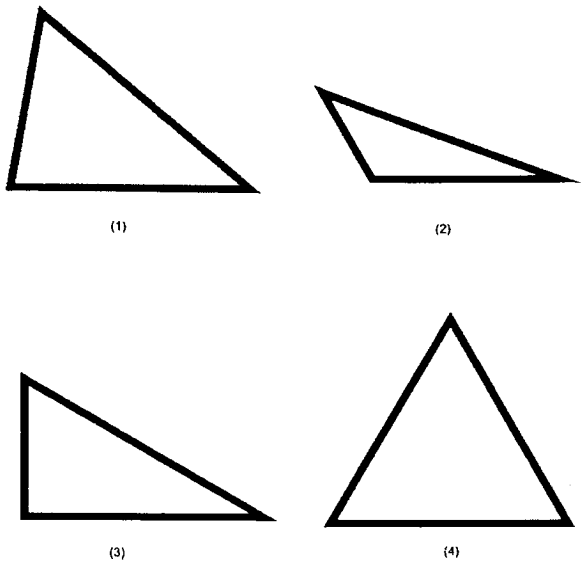
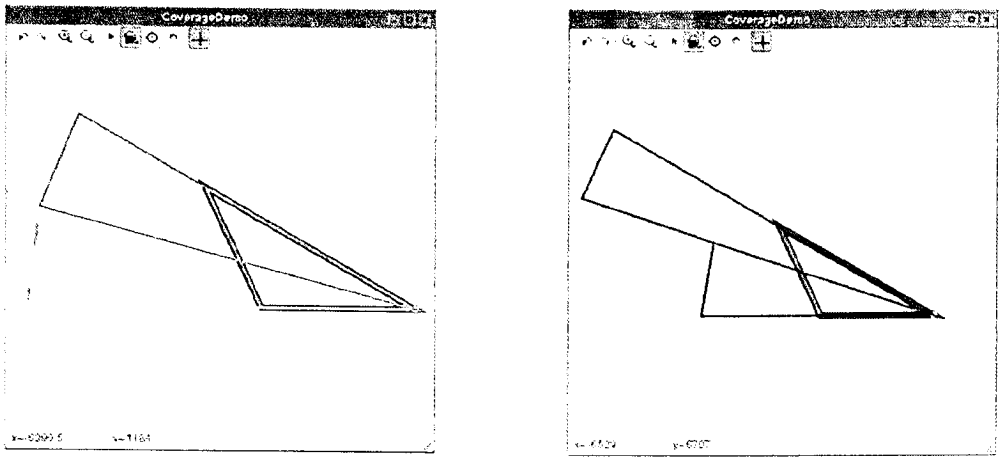


Figure 4.9: Types of Triangles: (1) acute, (2) obtuse, (3) right and (4) equiangular

a polygon may be partitioned. It was observed that different partitioning schemes offer various coverage benefits and that one method may be more suitable than another depending on the target coverage goals of the system. For example, using a midpoint partitioning scheme to partition a triangle, as opposed to an angle bisector scheme, results in creating larger variations in adjacent angles. This is illustrated in Figure 4.10. The angle bisector scheme creates even, relatively small angles which maximizes the use of a camera's area coverage ability, while the midpoint method creates unevenly distributed angles which may result in relatively large angles requiring more cameras to cover an area. On the other hand, variations in angles created by the midpoint scheme increase the likelihood that cameras will be placed in unique locations, thus making it more practical and robust.

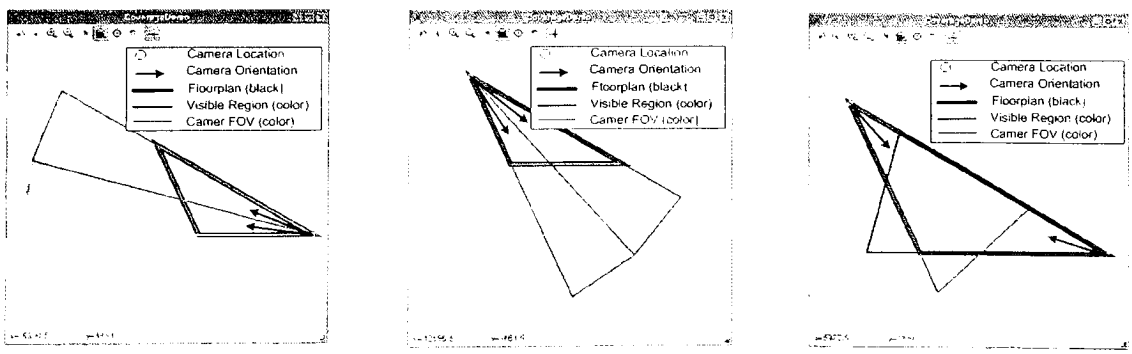
In addition to the scheme used for partitioning, there are also options for which angle to pick when partitioning a triangle. For instance, a triangle may be partitioned from its minimum (MIN), middle (MID) or maximum (MAX) angle. This is illustrated in Figure 4.11. Some observations to note are that in Figure 4.11(1), partitioning from the MIN angle results in creating triangles with relatively small angles to be covered, which maximizes the area coverage ability of the camera. Thus, this type of partitioning would typically



(1) (2)

Figure 4.10: Triangle Partitioning: (1) Angle Bisector vs. (2) Midpoint

require less cameras in general. In Figure 4.11 (3), we see that partitioning the MAX angle offers the advantage of breaking up the largest angle and thus partitions the longest edge of the triangle to be covered. The result would be more cameras with larger angles of view, thus we would need a greater number of cameras to cover a triangle. Also, note that MAX partitioning would create greater variations in the smallest angle of each triangular subsection, thus increasing the likelihood that cameras are placed in unique locations.



(1) (2) (3)

Figure 4.11: Triangle Partitioning: (1) MIN vs. (2) MID vs. (3) MAX

4.2.2 Adjustable Camera Parameters

Often, the initial camera placement algorithm located cameras in a manner that under-utilized the coverage capabilities of the camera. For example, consider the placement of the cameras in Figure 4.12. The algorithm places the cameras at the minimum angle of the triangles and sets the focal length accordingly to just cover the angle; however, this leads to a situation in which the maximum possible zoom level of the camera is used and the camera happens to be pointed at a wall. Any coverage ability offered by the camera that exceeds the distance to the wall goes unused.

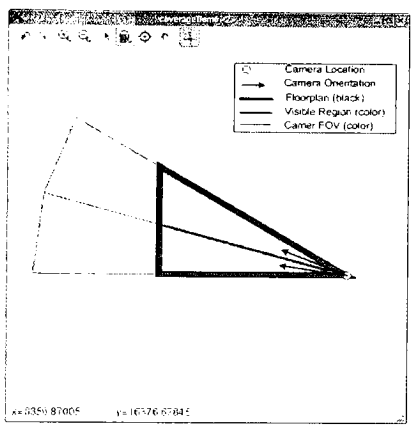


Figure 4.12: Coverage Utilization

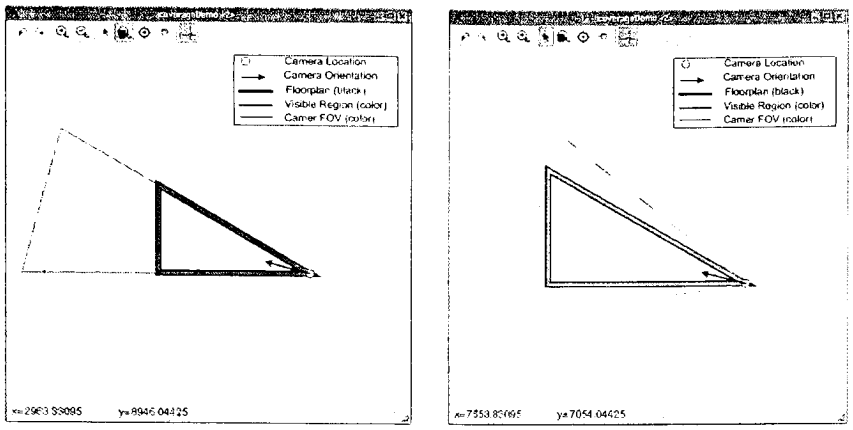
Depending on the shape of the room being covered, it may be possible to increase the coverage utilization of a camera by adjusting the zoom level of the camera. This would lead to increased overlap of the camera fields of view and thus and increase in network robustness. This concept is illustrated in Figure 4.13.

When considering dynamic coverage, cameras may also sweep over time, thus in this case one can also choose between sweeping as fast as possible to the greatest possible angle, or sweep at the minimum speed required to cover an angle.

The various combinations of settings for pan and zoom levels offer various advantages.

Case (I): Minimum Zoom, Minimum Pan

From the images in Figure 4.14, note that setting the pan and zoom levels to the minimum



(1) (2)

Figure 4.13: Adjustable Zoom, (1) Maximum zoom coverage and (2) Minimum zoom level coverage

required provides us with the ability to utilize all of the camera coverage area on one triangle. Note that there is no overlap of field of view into other regions. The net effect of this is that one can rotate the camera at the slowest possible required speed and limit the blind spot due to the application, represented in Figure 4.14 (1).

The slower rotational speed and reduction in effect of blind spot area due to the use of the minimum required pan and zoom levels will result in relatively greater practical coverage than the other cases of pan and zoom parameter adjustment.

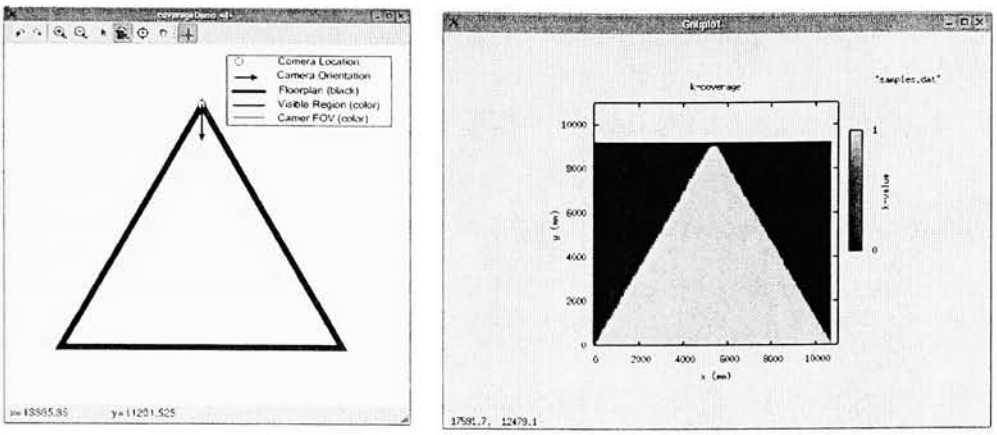
Case (II): Minimum Zoom, Maximum Pan

In Figure 4.15, observe that maximizing the pan ability allows us to overlap coverage into adjacent regions of the triangle to be covered. In addition, there is the benefit of smaller blind spot due to the use of the minimum focal length.

Case II will result in increased practicality due to the slower rotational angular velocity required as a result of the smaller focal length (larger angle of view). Also, this case will result in increased coverage due to the benefit of overlap into adjacent regions.

Case (III) Maximum Zoom, Minimum Pan

Figure 4.16 illustrates that fore Case III, maximum zoom allows us to cover an relatively



(1) (2)

Figure 4.14: Dynamic Camera Coverage: Case (I) Minimum Zoom, Minimum Pan. (1) Camera Placement and (2) K-coverage

larger amount of are than Case I and II which also overlaps into opposite regions. The minimum pan constraints the overlap to opposite regions. The trade off for increased coverage ability is the increase in uncovered area due to blind spot, as seen in Figure 4.16 (2).

This case offers increased efficiency due the ability to cover larger areas and increased robustness because of overlapping field of view into other regions, but will be overall less practical due to the amount of uncovered area.

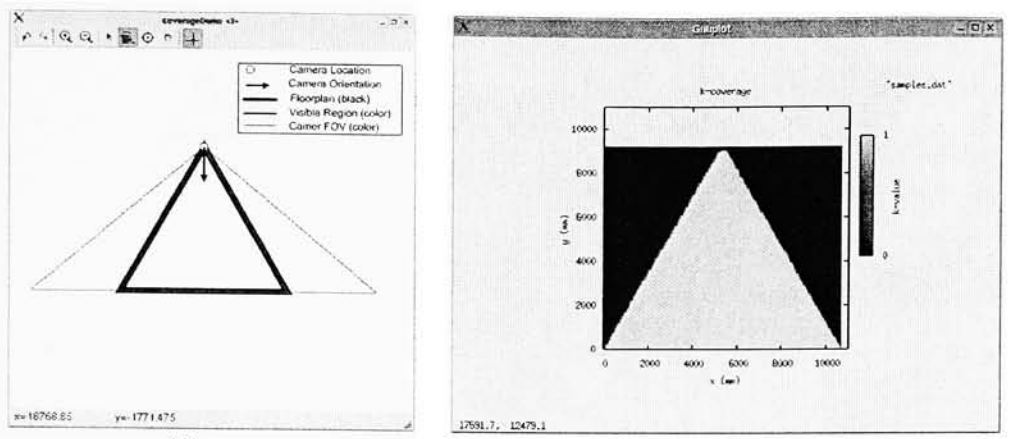
Case (IV): Maximum Zoom, Maximum Pan

By utilizing the maximum possible pan and zoom of the camera to cover a region, not in Figure 4.17 that one can cover the largest area and get the benefit of overlapping coverage to adjacent and opposite regions. The trade offs here are the increased blind spot area as well as the less practical coverage as it is required that the camera to rotate very fast.

This type of scenario will increase efficiency and robustness of a camera system, but is the least practical.

4.2.3 Restrictions on Camera Placement

In the initial camera placement algorithm, it is often the case that triangles are partitioned in a manner such that multiple cameras are placed in the same location. This is impractical



(1) (2)

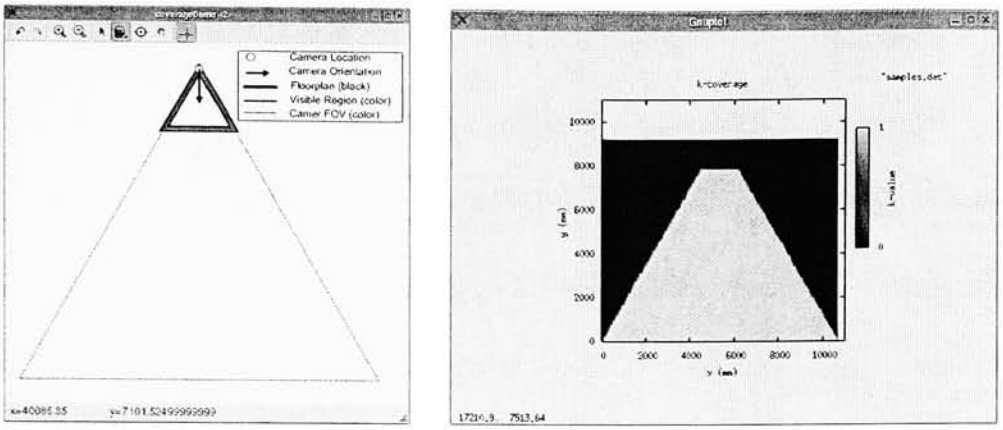
Figure 4.15: Dynamic Camera Coverage: Case (II) Minimum Zoom, Maximum Pan. (1) Camera Placement and (2) K-coverage

in real-world scenarios because there may be objects that occlude the field of view of a camera and one would like to avoid situations in which one object can occlude the view of multiple cameras. Figure 4.18 is an illustration of this.

To improve the coverage practicality of network of cameras, one can create situations in which there are multiple coverable angles to choose from and then choose between these angles based on whether or not a particular location already has a camera placed there. In other words, provide the ability to restrict camera placement to unique locations.

4.3 Strategies for Camera Placement and Parameter Adjustments

The identified critical variables for camera placement offer the ability to choose among various configurations for camera placement and camera parameter adjustment schemes to meet target performance goals. Observations from testing the various types of triangles led to the following strategies for improving performance relative to each of the identified metrics, namely (1) efficiency, (2) practicality and (3) robustness.



(1) (2)

Figure 4.16: Dynamic Camera Coverage: Case (III) Maximum Zoom, Minimum Pan. (1) Camera Placement and (2) K-coverage

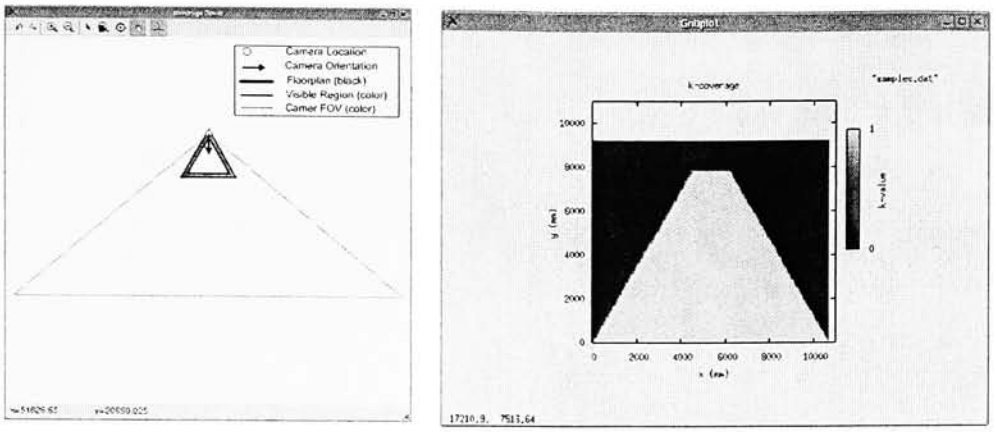
4.3.1 Efficiency

For efficient use of cameras, one is concerned with maximizing the use of the camera's area coverage ability and thus minimizing the number of cameras needed. The following summarizes the suggested options for efficient performance.

- **Partitioning**
  - **Angle Bisector**

    The angle bisector method allows for an even distribution of the angle between partitions, creating triangles with relatively small angles to be covered. This maximizes area coverage ability, thus requiring less cameras.
  - **MIN Angle**

    Again, this leads to triangles with relatively small angles of view and thus less cameras.
- **Camera Parameter Tuning**
  - **Maximize Zoom**



(1) (2)

Figure 4.17: Dynamic Camera Coverage: Case (IV) Maximum Zoom, Maximum Pan. (1) Camera Placement and (2) K-coverage

Maximizing the zoom level allows for overlap into opposite regions, thus efficiently utilizing the coverage ability of the camera.

- **Maximize Pan**

  Maximizing the use of the camera's pan capability gives us the ability to cover larger angles which will reduce the number of partitions and thus reduce the number of cameras needed.

4.3.2 Practicality

- **Partitioning**

  - **Midpoint**

    Midpoint partitioning is more likely to create larger variations in adjacent angles, thus increasing the likelihood that cameras will be placed in unique locations.

  - **MAX Angle**

    Again, partitioning from the MAX angle creates larger variations in adjacent angles and it is more likely for cameras to be placed in unique locations. In

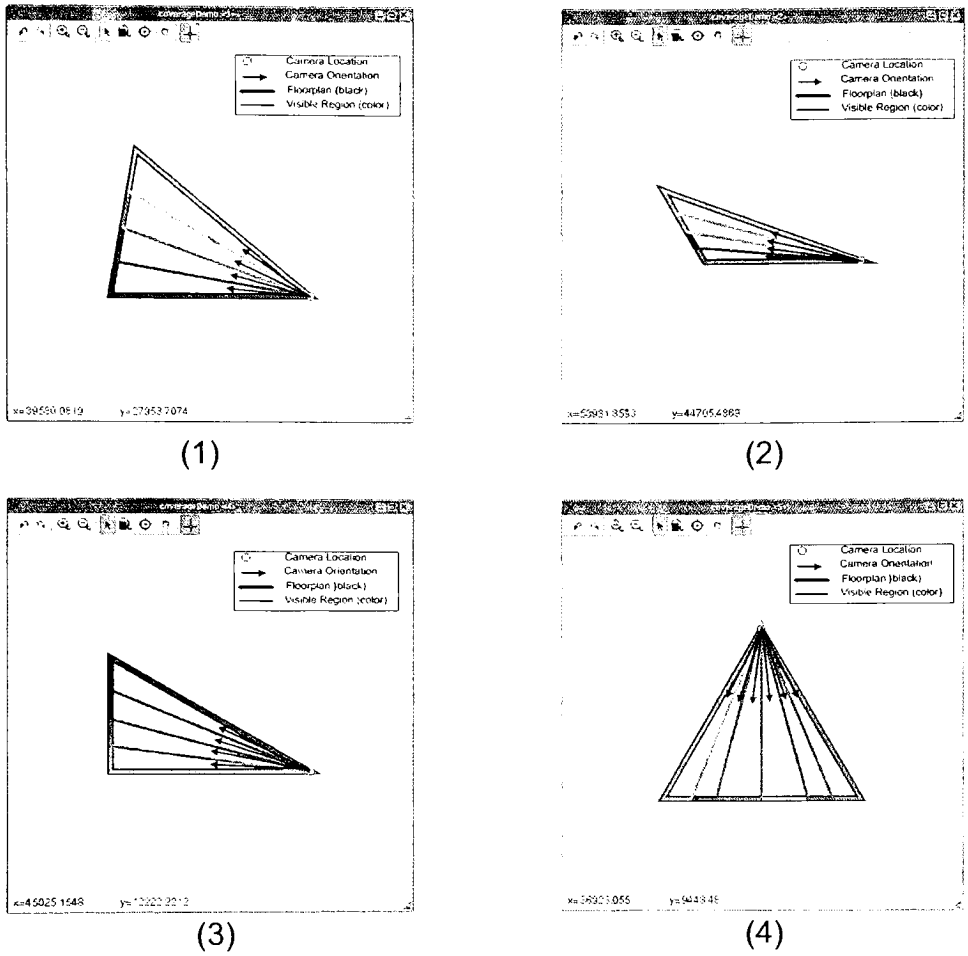


Figure 4.18: Practicality

addition, MAX angle partitioning also decreases the amount of uncovered area due to the blind spot caused by the minimum application distance requirement of the target application.

- **Camera Parameter Tuning**

  - **Minimize Zoom**

    Utilizing the minimum required zoom level to cover a triangle decrease the amount of uncovered area due to blind spot.

  - **Minimize Pan**

    Slower rotational speeds are more practical for real-world applications as image

processing algorithms can be sensitive to this factor. Minimizing the pan level refers to reducing the speed with which the camera needs to rotate to cover a particular angle to a minimum.

4.3.3 Robustness

- **Partitioning**
  - **Midpoint**

    As mentioned in the practicality section, midpoint partitioning offers greater variations in the adjacent angles to be covered. A camera system will be more robust to single points of failure if cameras are spread out in unique locations.
  - **MAX angle**

    MAX angle partitioning again offers the benefits of creating variations in adjacent angles. This increases the likelihood that cameras will have overlapping regions of coverage, or increased k-coverage.
- **Camera Parameter Tuning**
  - **Maximize Zoom**

    Maximizing the zoom level causes increased overlap into opposite regions and thus higher k-coverage.
  - **Maximize Pan**

    Maximizing the pan level causes increased overlap into adjacent regions and thus higher k-coverage.

4.4 Simulation Results

To show the effects of the camera parameter adjustment strategies, simulations were run on a sample set of real-world floor plan shapes and sizes. A few of the floor plans are

regular geometric shapes (triangle, square and hexagon) with real-world sizes. Others are examples from various related works and real-world floor plans (see [24] and [10]). Figure 4.19 illustrates the shapes of the sample rooms and Table 4.6 summarizes the number of sides for each N-gon and its area.

Floor Plan	N	Area
1	3	300
2	3	400
3	4	349.69
4	6	241.37
5	12	290
6	6	342
7	14	288
8	10	417
9	13	231.25
10	9	236

Table 4.6: Real-world Floor Plan Shapes and Sizes

Floorplan	# of cam	% cov per cam	% uncov per cam	% unique	K-cov	K-cov per cam
1	1	99.5306	0.0694	100.0000	0.9953	0.9993
2	1	99.8990	0.1010	100.0000	0.9990	0.9990
3	2	49.9540	0.0460	50.0000	0.9991	0.4996
4	4	24.9744	0.0256	100.0000	0.9990	0.2498
5	10	9.9921	0.0079	60.0000	1.1504	0.1150
6	4	24.9663	0.0337	75.0000	1.0367	0.2592
7	12	8.3275	0.0059	50.0000	1.0443	0.0870
8	8	12.4909	0.0091	62.5000	1.1385	0.1423
9	11	9.0756	0.0153	72.7273	1.0345	0.0940
10	7	14.2697	0.0160	57.1429	1.1962	0.1709
AVG	5.8889	37.7345	0.0349	74.4697	1.0445	0.3828

Table 4.7: Results from Real-world Floor Plan Shapes and Sizes: Partitioning

The first interesting simulation result is that in order to produce interesting data for the camera placement algorithm and camera parameter adjustment schemes (scenarios that actually utilize and take advantage of the improvements), one needs to overcome the hard limit set on the minimum number of cameras needed to cover a room due to the nature of the algorithm. Results for the initial choice from room shapes and sizes are tabulated in Table 4.7. In each, note that the room sizes are too small to obtain any meaningful info from

the camera placement and parameter adjustment schemes. We hit the best case coverage scenario for the algorithm every time. This limit is equal to the number of triangles created during the triangulation stage of the camera placement algorithm, which is equal to the number of sides of the polygon minus 2. Thus, there is a need to look at relatively large rooms sizes for the analysis of dynamic camera coverage with respect to the application of face detection.

To expose more interesting results, previously mentioned room sizes are scaled approximately 2-4x in the x and y directions. The resulting room shapes and sizes are tabulated in Table 4.8.

Floor Plan	N	Area
1	3	1452
2	3	1600
3	4	3147.21
4	6	2172.3282
5	12	2610
6	6	1368
7	14	2592
8	10	3753
9	13	3700
10	9	2124

Table 4.8: Real-world Floor Plan Shapes and Sizes

4.4.1 Angle Bisector vs. Midpoint Partitioning

Figures 4.20, 4.21, 4.22 illustrate the results of the various effects of different partitioning schemes on the real-world room samples. While our initial observations led us to believe that angle bisector and midpoint partitioning may offer some advantages over one another, we see from the simulation results that they offer comparable results in terms of efficiency, practicality and robustness. It is interesting to note though, that there are differences in some of the results. This leads to the idea that it may be worthwhile to explore other types of triangle partitioning schemes.

4.4.2 MIN vs. MID vs. MAX Angle Partitioning

The most appealing aspect of partitioning the minimum angle is that it creates smaller angles to be covered and thus needs smaller angles of view for coverage which in turn leads to larger focal lengths and greater area coverage ability.

Covering the smallest angle of view allows for the largest coverage area, but also produces the largest blind spot. The blind spot of the camera is caused by the minimum application distance required for an object to be within view of the camera. As zoom is increased, the angle of view of the camera is decreased and may cause target objects to go undetected at close distances in which the field of view is not wide enough to fit the entire object. The concept of Minimum Application Distance was illustrated earlier in Figure 2.7. Consider the use of the Sony EVI-D100 camera for Face Detection and plug into the equation for Minimum Application Distance. Note that in order for the entire face to be within the image (at a 1x zoom), a face needs to be at least 232.50 mm from the camera, which is larger than the specified Minimum Object Distance of 100 mm. This is summarized in Table 4.9.

Focal Length	Angle of View (H)	Minimum Application Distance
3.1 mm	54.60 deg	232.50 mm
31.0 mm	5.91 deg	2324.67 mm

Table 4.9: Minimum Application Distance (Sony EVI-D100 w.r.t Face Detection)

A very important observation that should be pointed out for the case of MIN angle partitioning is the practical limitation as room size increases. For smaller room sizes, one can take advantage of the adjustment of zoom in order to provide relatively larger coverage area for camera covering small angles. As the operational limits of the camera are approached though (limits of area that can be covered), note that there is a practical limit on how much coverage benefit can be extracted from the adjustment of camera parameters. This is especially meaningful when attempting to always partition triangles from the minimum vertex angle. This type of partitioning will cause the algorithm to never converge for larger rooms. For example, consider the examples presented in Figure 4.18. Notice that if the size of the

room were increased, the algorithm would continually partition the minimum angle trying to create a coverable triangle up to the point where the limit of the minimum angle we can cover is reached. A camera with maximum focal length would give us this minimum angle and would define the maximum area that can be covered by the camera. If the size of the room continued to be increased beyond this point, the algorithm would never be able to partition the room using this partitioning scheme alone.

This is due to the fact that the criteria for being able to cover a triangle requires that the largest side of the triangle can be covered by the side lengths of a camera field of view. So, the goal should be to create schemes for partitioning that target breaking up the longest side.

Partitioning the maximum angle offers two important benefits. First, it effectively eliminates the largest side from the coverage task. In addition, it reduces the largest angle into smaller sub components, producing more angles that may be coverable from a camera placement perspective.

Maximum angle partitioning is more likely to create larger angles in general, thus requiring cameras with larger angles of view that do not cover the same amount of area as cameras with smaller angles of view. This will require more cameras in general to cover the same amount of area. On the other hand, this also minimizes the blind spot produced due to the Minimum Application Distance requirement.

Maximum angle partitioning is more likely to produce situations in which camera locations are unique. By partitioning the largest angle, there is increased likelihood that the minimum angle in the newly created triangles is at a vertex other than the vertex of the angle that was just partitioned. Additionally, it is more likely that cameras will overlap, increasing robustness.

MID angle partitioning offers a balance between efficiency and robustness and would produce more practical camera layouts relative to MID angle partitioning for smaller rooms; however, this scheme also has the same limitation of MIN angle partitioning as room size

increases and the algorithm is not able to adjust camera parameters to extract more coverage.

4.4.3 Camera Parameter Tuning

Figures 4.23, 4.24, 4.25 illustrate the effects of camera parameter tuning on coverage in terms of the various performance metrics. Note that there is no improvement in terms of efficiency (Figure 4.23) and practicality (Figure 4.24) from tuning the camera parameters. This is due to the fact that the camera parameters are adjusted after the placement algorithm determines the location and orientation of the camera. Notice though that there are some significant advantages in terms of robustness when allowing for adjustments of the pan and zoom levels. Figure 4.23 illustrates an increase in coverage as first we are allowed to tune the zoom level and then both the zoom and the pan level. Therefore, it can be concluded that adjustment of camera parameters can be used, in addition to adjustments of parameters for partitioning and placement, to fine-tune coverage to meet target performance goals.

4.4.4 Restrictions on Camera Placement

The effects of restricting camera placement are illustrated in Figures 4.26 and 4.27 with respect to Floor Plan example 3.

An interesting observation of this enhancement is that it is largely dependent on the effectiveness of other enhancements in creating triangles with multiple coverable angles. For example, this enhancement will have no improvement when implemented with partitioning using the MIN angle, as there is no improvement on the likelihood of multiple coverable angles occurring within a triangle.

This improvement may be used in conjunction with other algorithm enhancement to improve the practicality and robustness of a camera system. Note that midpoint and maximum angle partitioning schemes increase the likelihood that the algorithm will create larger variations in angles during partitioning. Thus, the algorithm increases the likelihood of creating

triangles which have multiple coverable angles which can be exploited to place cameras in unique locations.

4.5 Limitations

One important limitation to note about the algorithm for camera placement is the hard limit set by the coarse-grained triangulation partitioning of the polygon. The algorithm depends on decomposing a room into triangles and the best case for triangulation is N-2 triangles, where N is the number of sides of the polygon.

Additionally, when increasing the size of the sample rooms, it was noticed that the MIN and MID partitioning schemes cause our algorithm not to converge for relatively large triangles. This is due to the fact that these schemes do not break up the longest edge of the triangle to be covered. For larger rooms, partitioning the maximum angle is the only option.

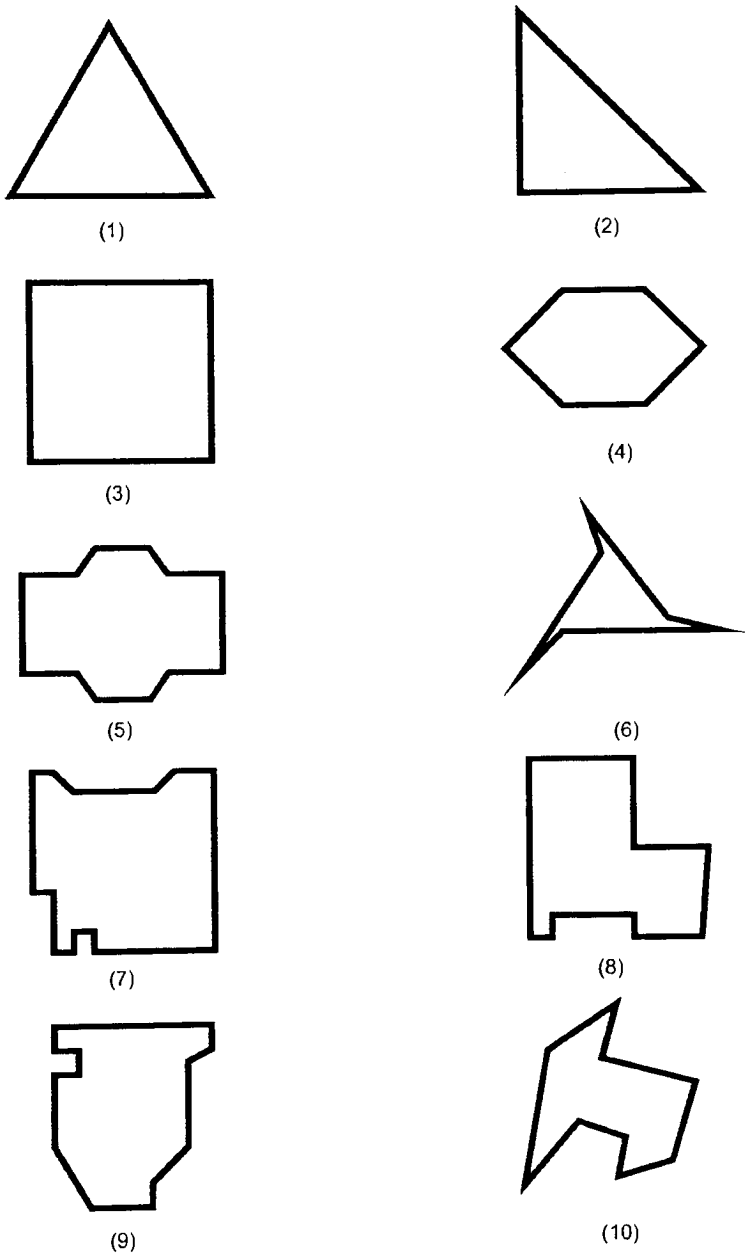


Figure 4.19: Real-world Floor Plan Shapes

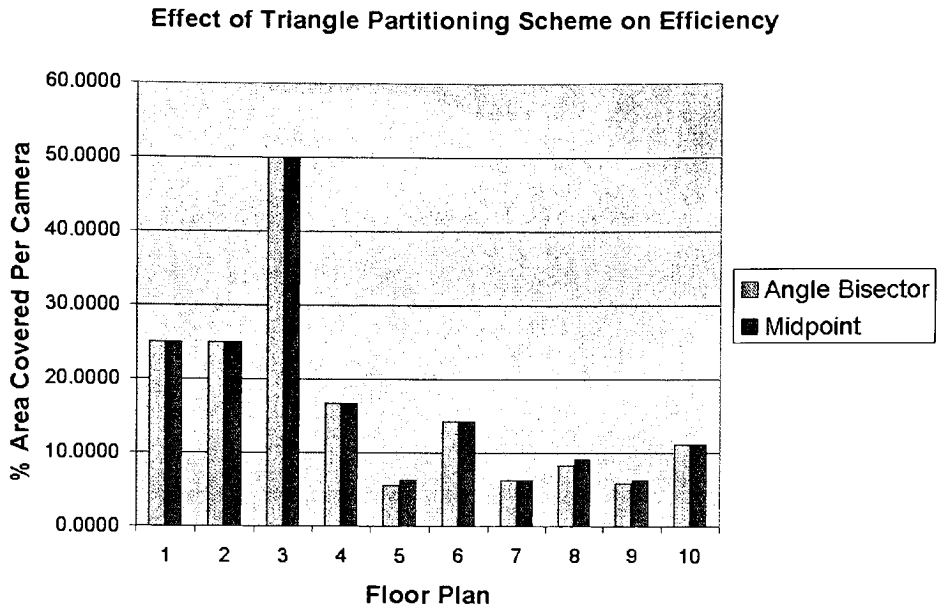


Figure 4.20: Effect of Triangle Partitioning Scheme on Efficiency

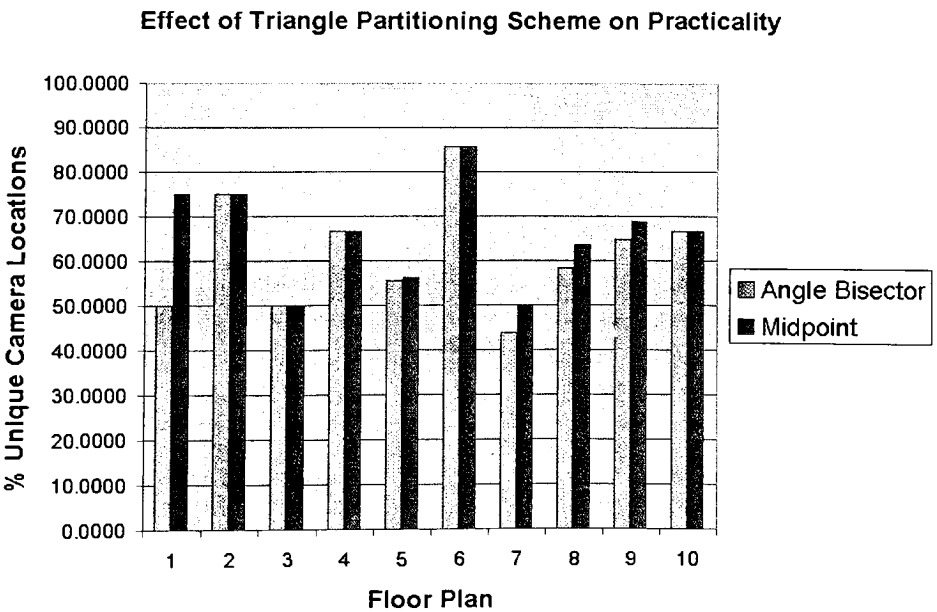


Figure 4.21: Effect of Triangle Partitioning Scheme on Practicality

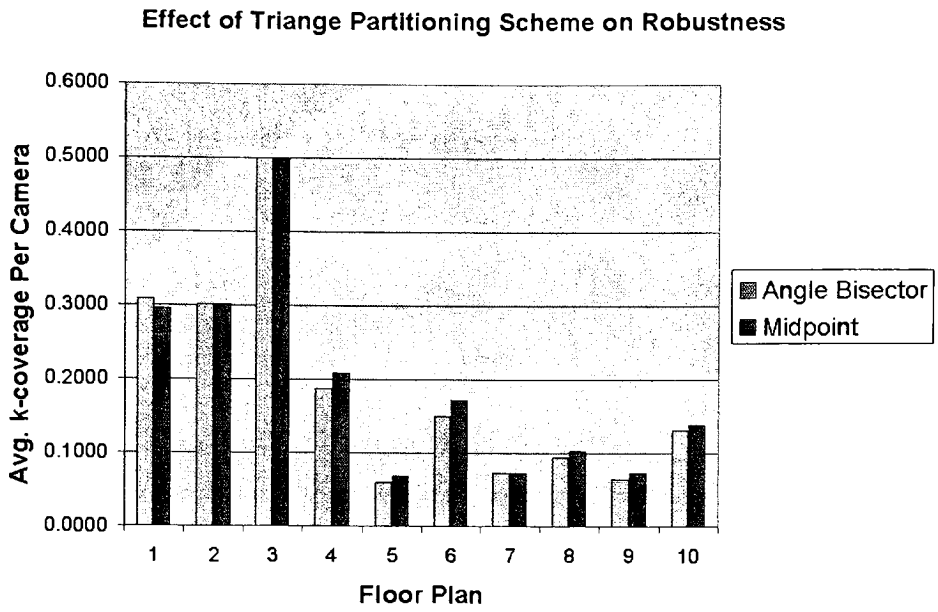


Figure 4.22: Effect of Triangle Partitioning Scheme on Robustness

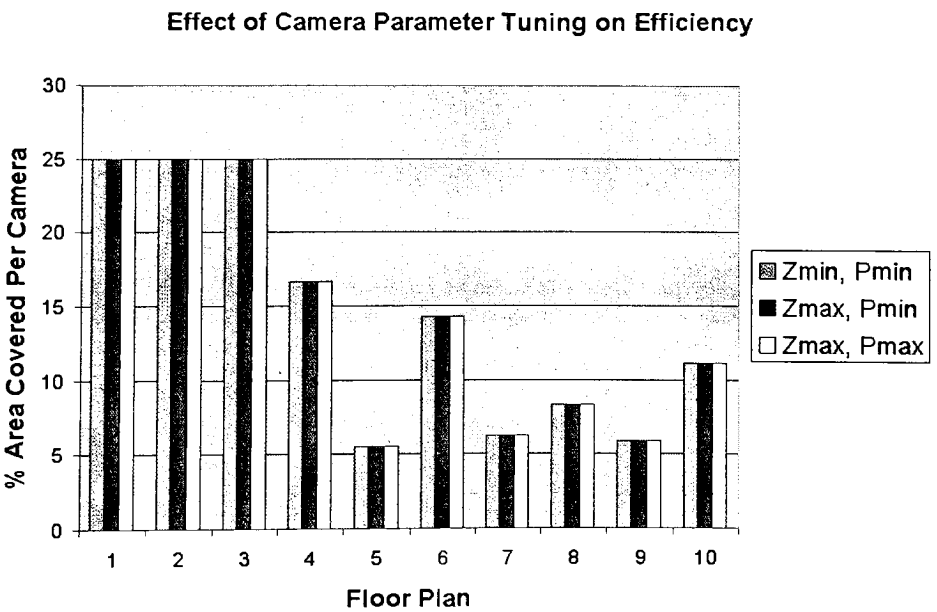


Figure 4.23: Effect of Camera Parameter Tuning on Efficiency

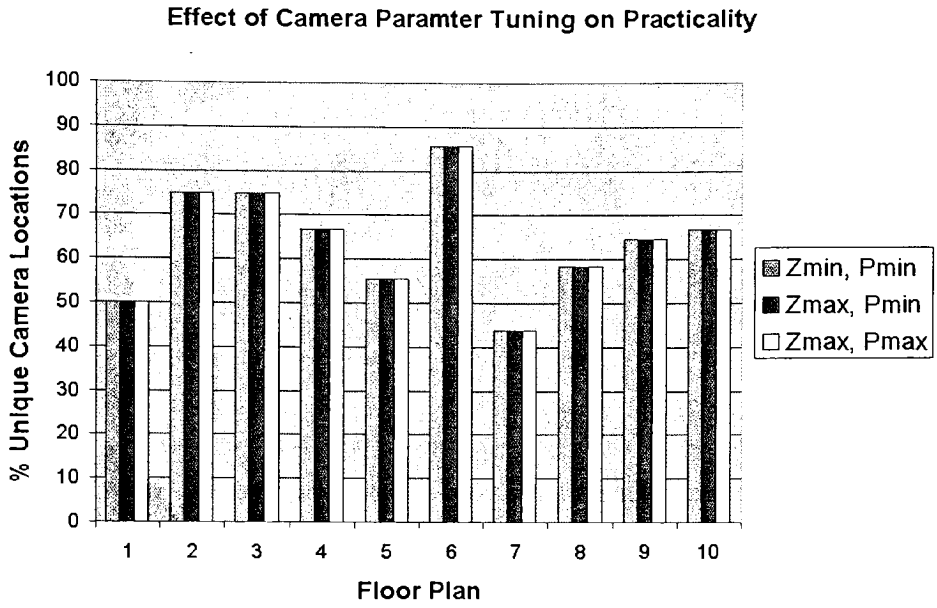


Figure 4.24: Effect of Camera Parameter Tuning on Practicality

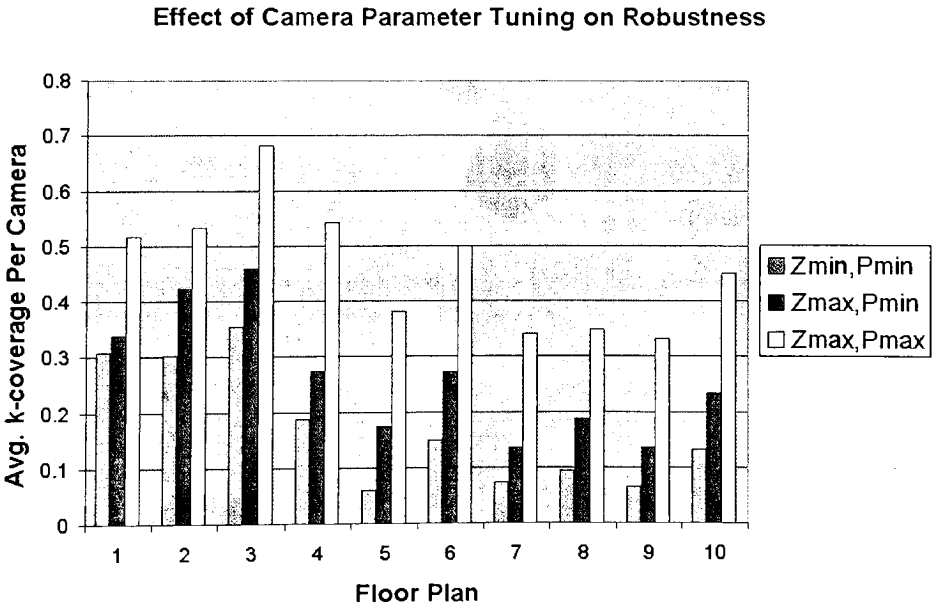


Figure 4.25: Effect of Camera Parameter Tuning on Robustness

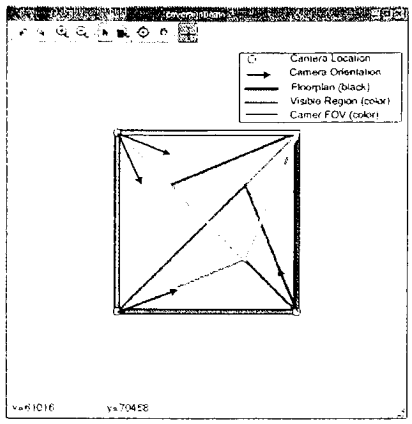


Figure 4.26: Camera Placement, no restrictions

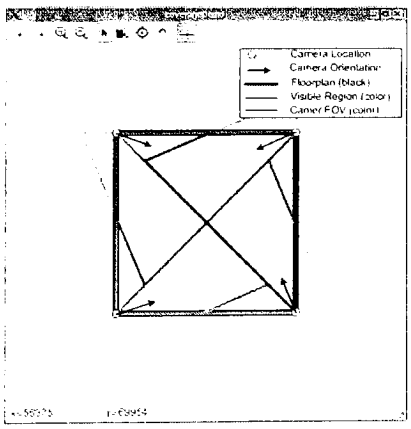


Figure 4.27: Restricted Camera Placement

Chapter 5

Concluding Remarks

Recall the goal of the work, which aims at determining an algorithm to place cameras in a manner that is efficient, practical and robust. Results from related work in the fields of camera placement and covering suggest that finding an optimal solution to such a problem may prove to be very difficult, if not intractable. Therefore, it can be concluded that a more meaningful contribution to the field of camera placement would be to attempt a heuristic approach and use it as a learning experience. The results of this work lead us to the following conclusions.

First, the camera placement algorithm for PTZ camera coverage provides a very practical and realizable approach to a very complex problem statement, *i.e.*, the novelty of this work lies in the simplicity of the approach. For instance, an important consideration when designing task-specific camera placement algorithms is the practical limitations of a camera, such as the limited field of view, directional sensing and limited pan and zoom speeds, and of the application, such as the minimum resolution requirements and the time constraints. Previous work has often either made unrealistic assumptions about the real-world constraints of a camera, such as infinite depth of field or infinite rotational speed, or have proposed overly complex models that do not lend themselves to the design of algorithms for camera placement. By building upon the approximation of the field of view of a camera as a triangle, this work has been able to represent practical limitations of a camera/application pair with relative simplicity. Also, this also for leveraging well understood principles of computational geometry.

Additionally, the implementation and analysis of the camera placement algorithm provides important insight on how to place cameras and adjust camera parameters to target various coverage goals. Recall the goals of efficient, practical and robust coverage. It has been found that one can develop strategies for camera placement and camera parameter adjustments to improve performance relative to each of these coverage goals. These strategies can be leveraged by future works that need to consider how to place cameras with respect to an image processing application with specific performance requirements.

Also note the performance of the camera placement algorithm is heavily dependent upon the size and the shape of a room and the partitioning scheme used to decompose the room into coverable components. This work considers partitioning a room into triangles and modeling the coverage of a camera as a triangle. This work follows a logical process of partitioning, attempting to cover the region, and repartitioning if the region cannot be covered. The results indicate that this may not be the most efficient way to divide a room. For instance, note that for the camera placement algorithm, there is a hard limit on the minimum number of cameras needed to cover a room based on the polygon triangulation. Various types of partitioning offer various benefits in terms of coverage goals. This is a point that may be exploited by camera placement algorithm designers to meet coverage performance goals of a system.

With the addition of a time constraint, the model for dynamic camera coverage can be reduced to static camera coverage. Again, the real novelty of this approach is its ability to reduce a highly complex problem, in this case one that spans time, into a relatively simpler problem that is more practical in terms of implementation. Since dynamic models for camera coverage are still in the early stages of being understood by researchers, this concept is particularly interest as it may serve as a building block for more complex camera placement algorithms.

5.1 Future Work

As noted in the concluding remarks, this work chose to represent the dynamic field of view of a camera as a triangle region because of the balance it provides between representativeness and tractability of the problem. It may be possible though to use more advanced models for camera coverage.

Also, this works makes many assumptions about the problem in order to reduce to a 2D formulation. A natural extension to this work would be to reintroduce the tilt parameter of the camera and extend these ideas to the 3D world.

Another area for extension of this work is the scheme for partitioning of the polygon regions. It was found that coverage performance is very dependent upon the type of partitioning scheme used. One could, for example, attempt to break the polygon region into a set of convex polygons before attempting to cover with cameras. Also, after partitioning, one could attempt to place a camera, and then recompute the amount of uncovered area after placing a camera and partition again. This would be useful for cases desiring to use the least number of cameras.

Another interesting idea would be to incorporate the camera field of view into the decision making process for partitioning schemes in order to do some intelligent partitioning of the room. This would create custom partitions tailored to the application and camera specifications.

The visibility algorithm provides a solution for occlusion due to walls of the polygon, but this work does not consider the scene to be changing over time. One idea would be to consider both dynamic camera coverage as well as a dynamic scene.

This work approaches the problem from a top-down perspective in which a relatively complex shape is broken down into simple subcomponents before placing cameras. Another idea would be to look at the problem from a bottom-up approach. Here, one could first consider simpler polygons and determine the best case for comparison purposes. Complexity could then be added to the problem as the subcomponents are better understood.

The approach taken in this work first checks whether or not a region can be covered,

then determines the visibility of the camera. A possible improvement would be to consider visibility in the decision of where to place cameras.

Finally, this work is limited to looking at simple polygons. Related works have extended coverage ideas to include solutions for polygons with holes. These holes could represent, occlusion due to objects within the room. This would add additional complexity to partitioning and placement though.

Bibliography

[1] Sachin Adlakha and Mani Srivastava. Critical density thresholds for coverage in wireless sensor networks. *IEEE Wireless Communications and Networking*, 3:1615–1620, March 2003.

[2] I. F. Akyildiz, E. Cayirci, Y. Sankarasubramaniam, and W. Su. Wireless sensor networks: a survey. *Computer Networks (Amsterdam, Netherlands: 1999)*, 38(4):393–422, March 2002.

[3] S. Banjeree and K. Kar. Node placement for connected coverage in sensor networks. In *Proceesings of Modeling and Optimization in Mobile, Ah-Hoc and Wireless Networks*. WiOpt 2003, March 2003.

[4] Sebastiano Battiato and Massimo Mancuso. An introduction to the digital still camera technology. *ST Journal of System Research*, 2(2):1–9, December 2001.

[5] Peter Brass, Benyuan Liu, Olivier Dousse, Phillippe Nain, and Don Towsley. Mobility improves coverage of sensor networks. In *Proceedings of ACM Mobihoc 2005*, volume 43, pages 67–73, Urbana-Champaign, Illinois, USA, May 2005.

[6] M. Cardei and D.Z. Du. Improving wireless sensor network lifetime through power aware organization. *Wireless Networks*, 11(3):333–340, May 2005.

[7] Mihaela Cardei and Jie Wu. Coverage in wireless sensor networks. Technical report, Florida Atlantic University, 2004.

[8] K. Daniels and R. Inkulu. Translational polygon covering using intersection graphs. In *Proceedings of the Thirteenth Canadian Conference on Computational Geometry*, pages 61–64, 2001.

[9] K. Daniels, A. Mathur, and R. Grinde. A combinatorial maximum cover approach to 2d translational geometric covering. *CCCG*, August 2003.

[10] Ugur Murat Erdam and Stan Sclaroff. Optimal placement of cameras in floorplans to satisfy task requirements and cost constraints. In *Proceedings of Omnivis2004, The fifth Workshop on Omnidirectional Vision, Camera Networks and Non-classical cameras*, pages 30–41, May 2004.

[11] Matthias Fichtner. A camera sensor model for sensor fusion. Master's thesis, Dresden University of Technology, October 2002.

[12] D. W. Gage. Command control for many robot systems. In *Proceedings of Nineteenth Annual AUVS Technical Synopsium*, pages 22–24, Huntsville, Alabama, USA, June 1992. AUVS-92.

[13] W.W. Gregg, W.E. Esaias, G.C. Feldman, R. Frouin, S.B. Hooker, C.R. McClain, and R.H. Woodward. Coverage opportunities for global ocean color in a multimission era. *IEEE Transactions on Geoscience and Remort Sensing*, 36:1620–1627, 1998.

[14] W. Haisheng and J. Zelek. A multi-classifier based real-time face detection system. *IEEE Trans. Robotics and Automation*, 2003.

[15] Chi-Fu Huang and Yu-Chee Tseng. The coverage problem in a wireless sensor network. In *Proceedings of WSNA*, pages 115–121, September 2003.

[16] Navitar Inc. Optical characteristics of video lenses. *Online publication accessible via http://www.navitar.com/zoom/cctv_op_char.htm*, 2005.

[17] Intel Open Source Computer Vision Library. *OpenCV Reference Manual.* available at http://www.intel.com/research/mrl/research/opencv/.

[18] Farinaz Koushanfar, Seapahn Meguerdichian, Miodrag Potkonjak, and Mani B. Srivastava. Coverage problems in wireless ad-hoc sensor networks. In *Proceedings of IEEE Infocom 2001*, volume 3, pages 1380–1387, April 2001.

[19] Farinaz Koushanfar, Seapahn Meguerdichian, Miodrag Potkonjak, and Mani B. Srivastava. Exposure in wireless ad hoc sensor networks. In *Proceedings of Annual International Conference on Mobile Computing and Networking (MobiCom '01),*, pages 139–150, July 2001.

[20] John J. Leonard and Jong Hwan Lim. Mobile robot relocation from echolocation constraints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(9):1035–1041, September 2000.

[21] Jun Lu and Tatsuya Suda. Coverage-aware self-scheduling in sensor networks. In *Proceedings of 2003 IEEE 18th Annual Workshop on Computer Communications*, pages 117–123, October 2003.

[22] Dinesh Manocha and Atul Narkhede. Fast polygon triangulation based on seidel's algorithm. *Online publication accessible via http://www.cs.unc.edu/ dm/CODE/GEM/chapter.html*, November 1995.

[23] A. Mittal and L.S. Davis. Visibility analysis and sensor planning in dynamic environments. In *European Conf. on Computer Vision (ECCV)*, 2004.

[24] Joseph O'Rourke. *Art Gallery Theorems and Algorithms*. Oxford University Press, Inc, 1987.

[25] Ashok Samal. Minimum resolution for human face detection and identification. Technical report, Department of Computer Science, University of Nebraska-Lincoln, 1991.

[26] Klamer Schutte. An edge labeling approach to concave polygon clipping. *Submitted to ACM Transactions of Graphics*, July 1995.

[27] Raimund Seidel. A simple and fast incremental randomized algorithm for computing trapezoidal decompositions and for triangulating polygons. *Computational Geometry: Theory and Applications*, 1(1):51–64, 1991.

[28] Sony Corporation. *Sony EVI-D100 Brochure*, 2002.

[29] The Imaging Source. An introduction to cameras. *Online publication accessible via http://www.theimagingsource.com/prod/cam/camintro_2.htm*, June 2003.

[30] The Imaging Source. An introduction to optics and lenses. *Online publication accessible via http://www.theimagingsource.com/prod/opt/opticsintro_2.htm*, June 2003.

[31] Eric W. Weisstein. Triangulation. *From MathWorld–A Wolfram Web Resource accessible via http://mathworld.wolfram.com/Triangulation.html.*

[32] Joseph W. Young. Head and face anthropometry of adult u.s. civilians. Technical report, Civil Aeromedical Institute, Federal Aviation Administration, 1993.